ESA 31
USER MANUAL

Electro Systems Associates Pvt Ltd
www.esaindia.com

PREFACE

This is the user's manual for ESA-31 microprocessor trainer. This manual describes the hardware and software components of ESA-31 and gives the interface information necessary for expanding the system.

This manual describes in detail the facilities offered by the Keyboard Monitor Program and Serial Monitor Program, the On-line Assembler, Disassembler packages. The on-board facilities; Audio cassette interface, Centronics parallel printer interface are also described in this manual. Communication with a Host Computer is also described.

Please note that this volume is a user's guide for ESA-31 and as such does not deal elaborately with the features of 8031 microcontroller and related peripherals and their programming. Details regarding these can be obtained from the following INTEL Publication.

MICRO CONTROLLER HANDBOOK

While every effort has been made to present the information in an accurate and simple fashion, we do welcome suggestions for improving the quality and usefulness of this manual.

Please address all your correspondence to :

ELECTRO SYSTEMS ASSOCIATES PVT LTD.,
4215 J.K. Complex, First Main Road, Subramanyanagar
P.O. Box No. 2139 BANGALORE - 560 021 INDIA
Fax : 91-80-3325615 Phone : 3323029 3322924
email : esaindia@vsnl.com,
www.esaindia.com

CONTENTS

CHAPTER 1 INTRODUCTION 1-1 to 1-3

CHAPTER 2 CONFIGURATION AND INSTALLATION 2-1 to 2-4

2.1 Configuration of ESA 31 2-1
2.1.1 Operational Mode Selection 2-1
2.1.2 Printer Enable/Disable 2-2
2.1.3 Baud Rate Selection 2-2
2.1.4 Memory Selection 2-2
2.2 Installation of ESA 31 2-3
2.2.1 Installation Procedure for Serial Mode of Operation 2-3
2.2.2 No Response in Serial mode 2-3
2.2.3 Installation Procedure for Keyboard Mode of Operation 2-4
2.2.4 No Response in Keyboard Mode 2-4

CHAPTER 3 KEYBOARD MONITOR 3-1 to 3-23

3.1 Introduction 3-1
3.2 RAM Usage 3-2
3.3 Keyboard and Display 3-2
3.4 Monitor Commands 3-2
3.4.1 Examine / Modify Memory Command 3-5
3.4.2 Examine / Modify Register Command 3-7
3.4.3 Go Command 3-9
3.4.4 Breakpoint Commands 3-11
3.4.4.1 Clear Break 3-11
3.4.4.2 Set Break 3-12
3.4.4.3 Display Break 3-13
3.4.4.4 Disable Break 3-14
3.4.4.5 Enable Break 3-15
3.4.5 Single Step Command 3-15
3.4.6 Block Move Command 3-17
3.4.7 Compare Command 3-18
3.4.8 Fill Command 3-20
3.4.9 Search Command 3-22

CHAPTER 4 SERIAL MONITOR 4-1 to 4-17

4.1 Introduction 4-1
4.2 Structure of monitor commands 4-1
4.3 Monitor commands 4-2
4.3.1 S (Modify Memory) Command 4-4
4.3.2 D (Display Memory) Command 4-5
4.3.3 M (Move Memory) Command 4-5
4.3.4 F (Fill Memory) Command 4-6
4.3.5 C (Compare Memory) Command 4-7
4.3.6 R (Examine/modify Registers) Command 4-7
4.3.7 J (Jump to Address/Set PC) Command 4-9
4.3.8 G (Go) Command 4-10
4.3.9 S (Single Step) Command 4-11
4.3.9.1 SR (Single Step with Register Display) Command 4-11
4.3.9.2 S (Single Step with Disassembly) Command 4-12
4.3.9.3 N (Single Step with Count) 4-12
4.3.10 B (Break point) Commands 4-13
4.3.10.1 Clear Breakpoint Command 4-13
4.3.10.2 Set Breakpoint Command 4-13
4.3.10.3 Display Breakpoint 4-14
4.3.10.4 Enable Breakpiont 4-14
4.3.10.5 Disable Breakpoint 4-15
4.3.11 A (Assembly Command) 4-15
4.3.12 Z (Disassembler Command) 4-16
4.3.13 H (Help Command) 4-17

CHAPTER 5 HARDWARE 5-1 to 5-10

5.1 Introduction 5-1
5.2 CPU, Address, Data and Control Signals 5-1
5.3 Memory Addressing 5-1
5.4 I/O Addressing 5-2
5.5 Keyboard / Display Interface 5-3
5.6 Programmable Interval Timer 5-5
5.7 Serial Interface 5-5
5.8 Programmable Peripheral Interface Devices 5-6
5.9 Programmable Input/output and Timer 5-6
5.10 Bus Expansion 5-6
5.11 Connector Details 5-7

CHAPTER 6 MONITOR ROUTINES ACCESSIBLE TO USER 6-1 to 6-2
6.1 Keyboard Monitor Routines Accessible to User 1
6.2 Serial Monitor Routines Accessible to User 2

CHAPTER 7 AUDIO TAPE INTERFACE 7-1 to 7-8
7.1 Introduction 7-1
7.2 Installation 7-1
7.2.1 Operational Hints 7-1
7.3 Operation from the Keyboard Monitor 7-2
7.3.1 Storing Data onto Tape 7-2
7.3.2 Reading Data from Tape 7-4
7.4 Operation from the Serial Monitor 7-5
7.4.1 Storing Data onto Tape 7-5
7.4.2 Reading Data from Tape 7-6
7.5 Data Recording Formats 7-7
7.5.1 Bit Format 7-7
7.5.2 Byte Format 7-7
7.5.3 File Format 7-7
7.6 Storage Capacity 7-8

CHAPTER 8 PARALLEL PRINTER INTERFACE 8-1 to 8-4
8.1 Introduction 8-1
8.2 Installation 8-1
8.3 Operation 8-2
8.4 Direct Output to Printer 8-2
8.5 Connector Details 8-3

CHAPTER 9 EPROM PROGRAMMER INTERFACE 9-1 to 9-10
9.1 Introduction 9-1
9.2 Installation Procedure 9-2
9.3 Operation from Serial Monitor 9-2
9.3.1 P Command 9-3
9.3.2 V Command 9-5
9.3.3 B Command 9-5
9.3.4 R Command 9-6
9.4 Operation from the Keyboard Monitor 9-7
9.4.1 PROG Command 9-8
9.4.2 VRFY Command 9-8
9.4.3 BLNK Command 9-9

9.4.4 PRRD Command 9-9
9.5 Examples 9-10

CHAPTER 10 COMMUNICATION WITH A HOST COMPUTER SYSTEM 10-1 to 10-8
10.1 Introduction 10-1
10.2 Installation 10-1
10.3 Returning to DOS 10-4
10.4 Operational details 10-4
10.4.1 Download operation 10-4
10.4.2 Upload operation 10-5
10.4.3 DOS Commands 10-7
10.4.4 Bottom Line 10-7
10.4.5 Command Recall 10-7
10.4.6 Communication 10-7
10.4.7 Help 10-8

CHAPTER 11 ASSEMBLER 11-1 to 11-15
11.1 Program to Display ESA P LTD in trainer Display 11-1
11.2 Program to Perform Multiplication of 2 numbers 11-2
11.3 Program to Perform division of 2 numbers 11-2
11.4 Program to Display
ELECTRO SYSTEMS ASSOCIATES PVT LTD BANGALORE
On the console 11-3
11.5 Program to convert ASCII to HEX equivalent on the console 11-4
11.6 Program to convert HEX equivalent to ASCII on the Display 11-4
11.7 Checking the 5th bit in the given byte 11-5
11.8 Program to display largest number among the 'N' numbers 11-5
11.9 Program to display decimal count 0 to 20 11-7
11.10 Program to display 24 hour digital clock in key board mode 11-8
11.11 Program to display 24 hour digital clock in serial mode 11-10

APPENDICES

APPENDIX A - COMPONENT PLACEMENT DIAGRAM
APPENDIX B - ASCII CODES
APPENDIX C - RS 232C CABLE REQUIREMENTS
APPENDIX D - PRODUCT LIST
APPENDIX E - INSTRUCTION SET

CHAPTER 1

INTRODUCTION

ESA31 is an extremely powerful microcontroller trainer based on the popular 8031 microcontroller. It can be used as a flexible instructional aid in academic institutions. Sophisticated features like Assembler and Disassembler allow the use of ESA31 for serious microcontroller developmental work in research institutions and R & D laboratories. On-board facilities like Audio Tape Interface, Serial interface (RS-232-C) and Centronics compatible Parallel Printer Interface enhance the power and utility of ESA31 system.

Following are the main features of ESA31:

- ★ ESA31 can be operated either from on-board keyboard or from a CRT terminal through its RS-232-C interface.
- ★ Keyboard and Serial Monitor Programs support the entry of user programs, editing, debug facilities like breakpoints and single-stepping, and full speed execution of user programs.
- ★ 1-Pass Assembler can assemble any memory-resident assembly language program.
- ★ 1-Pass Disassembler disassembles the object code into standard INTEL mnemonics.
- ★ On-board Audio Tape Interface provides MIC and EAR sockets for operation with an Audio Tape Recorder.
- ★ Total of 120KB memory is provided of which 64KB of memory is program memory and 56 KB of memory is data memory. The powerful monitor of the trainer occupies 32KB out of 64KB of program memory.
  There is a battery backup option for 32KB of program memory and 56KB of data memory.
  STD bus compatible signals available on the bus connector for easy expansion.

★ Driver Software for DOS, Windows facilitates downloading / uploading of user programs from / to a host computer system.

★ RS-232-C cable for serial interfacing.

Options Available

a. Interface Modules for training purpose (Calculator keyboard, Elevator, Display, ADC with DAC, Dual Slope ADC, 8-bit, 16 channel ADC, Dual DAC, Logic Controller, Crystal Clock Divider, Traffic Lights, RTC, Tone Generator, Stepper Motor, Numerical Printer, etc.)

b. 26 core ribbon cable connector set.

c. Ni-Cd batteries for battery backup.

SPECIFICATIONS :

MICROCONTROLLER : 8031 / 8051 operated at 11.0592 MHz

MEMORY : Four 28-pin JEDEC sockets offer 120KB of memory as follows: 32KB of firmware in one 27256 (program memory) 32KB of static RAM using one 62256 with optional battery backup as user program memory.

56KB of static RAM as data memory, using two 62256s with optional battery backup.

FIRMWARE : Serial and Keyboard Monitors,

1 pass Assembler , 1 Pass Disassembler Package (can be used in serial mode only)

Audio Tape Interface Driver Software

Centronics Printer Interface Driver Software

PERIPHERALS

8279-5 : To control 36 keys keyboard and 7-digit, 0.5" seven segment LED display.

8253-5: 3 Programmable Interval Timers, Timer 0,1,2 are available to the user.

8251 A: For serial communication, supports all standard bauds from 110 to 19,200. (Baud is selected through on board DIP Switch)

8255-2 : Two numbers available to user providing 48 programmable I/O lines

8155-2: Used by the system to read DIP Switches, to control centronics printer ,audio tape interface and the timer is used to generate baudclock for serial communication and 128 bytes of RAM is used as a scratch pad memory.

BUS EXPANSION

STD bus compatible signals are available on a 50 pin right angle connector.

INTERFACE SIGNALS

Parallel I/O :	48 lines (2 X 8255-2) of TTL compatible bus brought out to two spectra-strip type ribbon cable connectors.
Serial I/O :	RS-232-C with standard MODEM control signals through on board 9 pin D-type female connector.
Cassette interface signals :	Available on MIC and EAR sockets
Parallel printer signals :	Centronics compatible parallel printer interface signals available on a 25 pin D - type female connector in PC compatible pin configuration.

POWER SUPPLY (Optional)

+5V, (±0.1V), 3A

CHAPTER 2

CONFIGURATION AND INSTALLATION

2.1 CONFIGURATION OF ESA31

ESA31 Microcontroller Trainer is versatile and can be configured in a number of ways as determined by the settings of a DIP Switch (refer to the component layout diagram in Appendix A to locate the DIP Switch). This chapter describes all the configuration options and the installation procedures.

2.1.1 OPERATIONAL MODE SELECTION

ESA31 can be operated either in the hexadecimal keyboard mode or in the serial mode. In the hexadecimal keyboard mode the trainer is operated from the hexadecimal keyboard/display unit. In the serial mode, the trainer is connected to a CRT terminal or to a host computer system (like PC Compatibles) through an RS-232-C interface. In either mode of operation, the system provides a variety of commands for program development/debugging. However, several features of ESA31 like on-line assembler, disassembler etc., are available only in the serial mode of operation. The selection of the desired mode of operation is done as follows:

Switch 4 of the DIP Switch	Operational Mode
OFF	Hexadecimal Keyboard Mode.*
ON	Serial Mode

(* factory installed option)

Chapter 3 describes the commands available in keyboard mode and chapter 4 describes the commands available in serial mode.

2.1.2 PRINTER ENABLE/DISABLE

ESA31 firmware includes the driver program for centronics compatible parallel printer interface. This driver can be enabled/disabled as shown below:

Switch 5 of the DIP Switch	Printer driver
OFF	disabled. *
ON	enabled.

(* factory installed option)

Chapter 8 describes the parallel printer interface in detail.

2.1.3 BAUD RATE SELECTION

In the serial mode of operation, ESA31 configures the on-board 8251A USART as follows:

* asynchronous mode
* 8 bit character length
* 2 stop bits
* no parity

Timer of the on-board 8155-2 provides the transmit and receive baud clocks for the USART. (Refer chapter 5 for a detailed discussion of the hardware). This timer is initialised by the system firmware to provide proper baud clock based on the settings of the DIP Switch at the time of Reset or Power-On as shown below:

Baud Rate Selection

S3	S2	S1	Baud Rate
ON	ON	ON	110
ON	ON	OFF	300
ON	OFF	ON	600
ON	OFF	OFF	1200
OFF	ON	ON	2400
OFF	ON	OFF	4800
OFF	OFF	ON	9600*
OFF	OFF	OFF	19,200

(* factory installed option)

2.1.4 MEMORY SELECTION

ESA31 has four 28-Pin sockets for memory. System firmware (32K bytes) is supplied in a 27256 EPROM at the socket U8. 32K bytes of static RAM is provided by a 62256 at the socket U9 as Program Memory. 32K bytes of Data Memory is provided at U10 as Data Memory. The fourth socket at U11 is populated with 62256 to provide, 24K bytes of Data Memory.

2.2 INSTALLATION OF ESA31

To install ESA31, the following accessories are required.

a) Power Supply 5V, 3 Amp

b) For Serial mode of operation :

CRT terminal with RS-232-C interface or host system (like an PC Compatible) with the driver software for host system. (Refer chapter 10 for details).

2.2.1. INSTALLATION PROCEDURE FOR SERIAL MODE OF OPERATION

a) Select serial mode of operation (Ref. Section 2.1.1)

b) Select printer if required (Ref. Section 2.1.2)

c) Set the desired baud rate (Ref. Section 2.1.3)

d) Connect ESA31 to the CRT terminal/host system through RS-232-C cable (Appendix C describes the RS-232-C interface requirements) over the connector J7. (Refer Appendix A for locating the connectors). If a terminal is being used, turn-on the terminal. If a computer system is being used, turn-on the system and execute the driver software (Ref. Chapter 10 for details).

e) Connect the power supply of required capacity to ESA31 and turn-on the power.

Power On Self Test will be done. Following initialisations are also done.

All registers will be initialised to reset condition of 8031/8051. Breakpoints in both program and data memory will be cleared and disabled. Then appears the Sign On message as follows.

ESA-8051 Serial monitor V x.y

(V x.y indicates version x and revision y)

The sign-on message is followed by the command prompt, “>” in the next line. Now ESA31 is ready for operation in serial mode.

NOTE : The keyboard/display module will display “Serial”

2.2.2 NO RESPONSE IN SERIAL MODE

If there is no response from ESA31 in serial mode, after installing it as described in the previous section, check the following items:

a) Check the RS-232-C cable connections at both the ends. (Appendix C describes the interface in detail)

b) Check the power supply connections and voltage levels.

c) Check the handshake signals of RS-232-C interface (Ref.Appendix C)

d) Check the baud rates of ESA31 and the device connected to it.

e) If a computer system is the controlling device,check that the driver program is running, the RS-232-C cable is connected to the correct port and the port is working

f) Check the configuration of ESA31 again. (DIP Switch settings)

NOTE:

DIP Switch status is read only at Power-On / Reset. If you change the settings, either press the RESET key or switch OFF and then switch ON the power supply. If the problem still persists, please contact the manufacturer.

2.2.3. INSTALLATION PROCEDURE FOR KEYBOARD MODE OF OPERATION

a) Select Keyboard mode of operation (Ref. Section 2.1.1)

b) Connect the power supply of required capacity to ESA31 and switch ON the power.

c) Now the following message appears on the address field of seven segment display for few seconds.

POST

Power-ON Self Test will be done. Following initialisations are also done.

All registers will be initialised to Reset condition of 8031/8051.

Breakpoints in both program and data memory will be cleared and disabled.

Then appears the sign on message as follows.

- ESA 51

Now ESA 31 is ready for operation in the keyboard mode.

2.2.4 NO RESPONSE IN KEYBOARD MODE

If the correct sign-on message does not appear in the keyboard mode, check the following items.

a) If the seven-segment display is blank, check the power supply connections and voltages.

b) If the seven-segment display shows random pattern, check the configuration settings once again.

NOTE: DIP Switch is read only at Power-On/Reset. If you change the settings, either press the RESET key or switch OFF and then switch ON the power supply.

If the problem persists, please contact the manufacturer.

CHAPTER 3

KEYBOARD MONITOR

3.1 INTRODUCTION

This chapter describes the commands supported by the keyboard monitor program. In the keyboard mode, the user enters the commands and data by pressing the appropriate keys on the keypad. Responses are displayed by the system on the seven-digit LED display.

Whenever the monitor expects a command, the display shows a dash("-") at the left edge of the address field, possibly along with an error message or with the sign-on message upon reset. When the monitor expects parameter(s), decimal point(s) will be displayed on the field into which the parameter(s) is (are)to be placed. The parameter will be either an address to be entered in the address field or a byte of data to be entered in the data field (The only exception, explained later, occurs in the use of the EXAM REG command). The valid range for an address parameter is from 1 to 4 hexadecimal digits and the valid range for a data parameter is from 1 to 2 hexadecimal digits. Longer numbers may be entered but such numbers are evaluated modulo 64K or 256 respectively, i.e. only the last four or the last two digits entered will be accepted.

The RESET key causes a hardware reset and restarts the monitor. The monitor displays the sign-on message (-ESA51) across the address and data fields of the display. Now the monitor will be ready to accept the commands from the user. But the monitor does not save the information about the state (register values etc) of any previous user program. However, contents of the user portion of the RAM area are not disturbed by the monitor.

3.2 RAM USAGE

The system monitor utilizes 256 bytes of data memory, from E000H to E0FFH as reserved area for system. User programs should not disturb this area, otherwise the results are unpredictable.

3.3 KEYBOARD AND DISPLAY

As noted already, the display consists of 7 seven-segment LED displays, separated into three fields. The left most single digit forms the special field, next four digits form the address field and the last two digits form the data field.

The 36 key keypad consists of the following groups of keys.

a. **Hexpad:** 16 keys representing the hexadecimal digits 0 through F. In the use of the EXAM REG command, some of these keys represent register names also, as indicated by the legends on the keytops. This usage is further explained in the description of the EXAM REG command.

   Further, two of these keys serve the functions of command keys also. The context of operation defines the meaning of the key.

b. **Command Group:** 13 command keys of which 2 keys (DPBRK & SERCH) are delimiter keys also. These commands are in addition to the two commands mentioned above.

c. **Memory Group:** 4 keys (PRG MEM, EXT DATA, BIT MEM, INT DATA) to select the type of memory.

d. **System Operation Keys:** RESET, BREAK and EXEC keys. RESET key, as already noted causes a hardware reset of the system. BREAK can be used to stop the execution. EXEC is used to invoke the execution of commands and to terminate commands.

3.4 MONITOR COMMANDS

The keyboard monitor is capable of executing fifteen individual commands, summarized in Table 3.1. Each command is represented by exactly one key with appropriate legend on the keytop. The commands are described in detail in the sections which follow. In both the table and in the individual command descriptions, the following notation is used:

Upper case letters and numbers represent keyboard keys:

NOTATIONAL CONVENTIONS

SYMBOL	NAME	USAGE
{}	curly braces with ellipses	Encloses a required argument 1 or more times.
[]	square brackets	encloses an item that appears 0 or 1 time.
[]....	square brackets	encloses an item that appears 0 or more times
	vertical bar italics	seperates alternative items in a list. indicates a descriptive item that should be replaced with an actual item.

TABLE 3.1 KEYBOARD MONITOR COMMAND SUMMARY
COMMANDS & FUNCTION/FORMAT

EXAMINE/MODIFY MEMORY:

Displays/Modifies the contents of a memory location.

EXAM MEM {PRG MEM | EXT DATA | BIT MEM | INT DATA} addr1 NEXT [[[data] NEXT| PREV]...] EXEC

EXAMINE/MODIFY REGISTER:

Displays/modifies 8031 / 8051 register contents.

EXAM REG {reg key} [[[Data] NEXT | PREV]...] EXEC

SINGLE STEP:

Executes a single user program instruction.

SINGLE STEP [start addr] [NEXT count] EXEC [NEXT ...] EXEC

GO :

Transfers control from monitor to user program

Go [addr] EXEC

BLOCK MOVE:

Moves a block of data from one portion to another.

BLK MOVE {PRG MEM | EXT DATA | INT DATA} start addr NEXT end addr NEXT {PRG MEM | EXT DATA | INT DATA} dest addr EXEC

SET BREAKPOINT:

Sets a breakpoint in program memory or data memory with facility of range (ORing).

SET BRK {PRG MEM | EXT DATA} addr [[[NEXT]...] EXEC]
[PREV <range -end addr>] EXEC

CLEAR BREAKPOINT:

Clears a breakpoint in program memory or data memory with facility of range (ORing).

CLR BRK {PRG MEM | EXT DATA} addr [[[NEXT]...] EXEC]
[PREV <range-end addr>] EXEC

ENABLE BREAKPOINT:

Enables breakpoint in program memory or data memory.

EABLE BRK {PRG MEM | EXT DATA} [NEXT {PRG MEM | EXT DATA}] EXEC

DISABLE BREAKPOINT:

Disables the breakpoint in program memory or data memory.

DABLE BRK {PRG MEM | EXT DATA} [NEXT {PRG MEM | EXT DATA}] EXEC

COMPARE MEMORY:

Compares two blocks of memory

COMP {PRG MEM | EXT DATA | INT DATA} start addr NEXT end addr NEXT
{PRG MEM | EXT DATA | INT DATA} dest addr EXEC

FILL MEMORY:

Fills a block of memory with a byte constant

FILL {PRG MEM | EXT DATA | INT DATA} start addr NEXT end addr NEXT data EXEC

SEARCH:

Searches a string of data in program or data memory

SERCH {PRG MEM | EXT DATA | INT DATA} start addr NEXT end addr
NEXT data1 [[[NEXT data2] NEXT data3] NEXT data4] EXEC

DISPLAY BREAKPOINTS:

Displays the breakpoints in program memory or data memory.

DPBRK { PRG MEM | EXT DATA} NEXT EXEC

TPRD: Reads from the tape

TPRD file name EXEC

TPWR: Write to the tape

TPWR file name NEXT {PRG MEM | |EXT DATA}
start addr NEXT end addr EXEC

NOTE: Whenever a dot appears in the special field user has to enter the type of memory by pressing the following keys as the case may be.

Type of memory to be selected	Key to be pressed
Program memory	PRG MEM
External data memory	EXT DATA
Bit memory	BIT MEM
Internal data memory	INT DATA

3.4.1 EXAMINE/MODIFY MEMORY COMMAND

FUNCTION

This command is used to examine the contents of selected memory locations. The contents can be optionally modified if the memory location is in RAM area.

FORMAT

EXAM MEM {PRG MEM | EXT DATA | BIT MEM | INT DATA} addr1
NEXT [[[data] NEXT | PREV]...] EXEC

OPERATION

1. To use this command, press the EXAM MEM key when prompted for a command. When this key is pressed, the display is cleared and a dot appears at special field prompting the user to enter the type of memory to be examined. Enter the type of memory by pressing PRG MEM, EXT DATA, BIT MEM or INT DATA key. A dot appears at the last digit of the address field indicating that an address entry is required.
2. Enter the memory address of the byte to be examined. (memory addresses are evaluated modulo 64K if it is program memory or data memory and modulo 256 if it is internal data memory or bit memory) This value is displayed in the address field of the display.
3. After entering the address value, press the NEXT key. The data byte at the addressed memory location will be displayed in the data field and a decimal point (a dot) appears at the right edge of the data field indicating that data may be updated.
4. If the contents of the addressed memory location are only to be examined, press the EXEC key to terminate the command or press the NEXT key to examine/modify the next consecutive memory location or the PREV key to examine/modify the previous memory location.
5. To modify the contents of an addressed memory location, enter the new data from the keypad (note that data values are evaluated modulo 256). However, this new value, displayed in the data field is not entered into the addressed memory location till NEXT or EXEC or PREV is pressed.

ERROR CONDITIONS:

Error conditions occurs while attempting to modify a non existent or Read-Only Memory (ROM) location. Note that the error is not detected until the PREV or NEXT or EXEC key is pressed. When an error is detected, the characters ‘Err’ are displayed along with the command prompt character (-) in the address field.

EXAMPLES:

Example 1: Examining a series of memory locations starting from 0000H (the start of keyboard monitor).

Key Pressed	Display			Comments
	Special Field	Address Field	Data Field	
RESET		ESA	51	System Reset
EXAM MEM	-			Examine Memory Command
PRG MEM	P	.		Enter type of memory
0	P	0000.		First memory location to be examined
NEXT	P	0000	02.	Contents of this location
NEXT	P	0001	00.	Next location and its contents
NEXT		0002	30.	Next location and its contents
PREV	P	0001	00.	Previous location and its contents
EXEC		-		Command termination / prompt

Example 2 : Examine and modify the contents of memory location 8D00H.

Key Pressed	Display			Comments
	Special Field	Address Field	Data Field	
RESET		-ESA	51	System Reset
EXAM MEM	.			Examine Memory Command
PRG MEM	P			Type of memory
8		0008.		
D	P	008d.		Memory location to be examined & modified
0	P	08d0.		
0	P	8d00		
NEXT	P	8d00	XX.	Contents of this location
A	P	8d00	0A.	New data to be entered
F	P	8d00	AF	
EXEC		-		Command termination prompt after updating the data.

To check that data was updated successfully, press EXAM MEM key, program memory as option and enter memory address 8D00H and note that “AF”is displayed in the data field when NEXT key is pressed.

Example 3: Trying to modify ROM location.

Key Pressed	Display			Comments
	Special Field	Address Field	Data Field	
RESET		-ESA	51	System Reset
EXAM MEM	-			Examine Memory Command.
PRG MEM	P	-		Type of memory
A	P	000A		Address of location to be examined.
NEXT	P	000A	FF.	contents of location.
7	P	000A	F7.	
A	P	000A	7A.	New data to be entered
NEXT		-Err		Error message

You tried to modify the contents of a Read Only Memory location. Hence, the error message along with command prompt character was displayed. You can repeat the above sequence of keys to see that the contents of this location are unaltered.

3.4.2 EXAMINE / MODIFY REGISTER COMMAND

FUNCTION

The examine register command is used to examine and optionally modify the contents of some of the 8031/8051’s registers.

FORMAT

EXEM REG {reg key} [[[Data] NEXT | PREV]...] EXEC

OPERATION

1 To use this command, press the EXAM REG key when prompted for a command. Now, the display is cleared & ‘rE’ is displayed in the address field. Thus the next hexadecimal key pad entry will be interpreted as a register name, if valid.

TABLE 3.2. PROCESSOR REGISTERS

Register Name	Display
Register A	A
Register B	b
Stack pointer	SP

Register Name	Display
Flags Register	F
Data pointer high	dPH
Data pointer low	dPL
Register TH0	tH0
Register TL0	tL0
Register TH1	tH1
Register TL1	tL1
Register P1	P1
Register P3	P3
Program counter high	PCH
Program counter low	PCL
Register R0	r0
Register R1	r1
Register R2	r2
Register R3	r3
Register R4	r4
Register R5	r5
Register R6	r6
Register R7	r7

FORMAT OF THE FLAG BYTE F:

C	AC	F0	RS1	RS0	OV	-	P
Carry Flag	Auxillary Carry Flag	User Flag	Bank Select 1	Bank Select 0	Overflow Flag	User Flag	Parity Flag

Fig 3.1. : FORMAT OF REGISTER

2. When the hexadecimal key is pressed, the corresponding register abbreviation is displayed in the address field and the contents of this register are displayed in the data field and a data update prompt (dot) appears at the right edge of the data field.

   Table 3.2. defines some of register names, the hexadecimal keyboards acronyms, the abbreviations appearing in the address field of the display and the sequence in which the registers are displayed.

3. When the register contents are displayed (with the data update prompt), the contents of this register can be modified if desired. To do this, enter the new value from the keyboard. This new value will be displayed in the data field and the register contents are updated when either the PREV or NEXT or EXEC key is pressed.

4. After examining and optionally modifying the contents, if the EXEC key is pressed, the command is terminated. If the NEXT key is pressed, contents of the “next register” (according to the order

shown in Table 3.2) are displayed and opened for modification. Note that the sequence is cyclic and thus pressing the NEXT key when the R7 is displayed will not terminate the command. If the PREV key is pressed, the contents of the "previous register" (according to the order shown in Table 3.2) are displayed and opened for modification.

5. When any of the registers R0-R7, has to be examined, press EXAM REG key and then BIT MEM key. Now press keys 0-7 on the hexadecimal keypad, which corresponds to register R0 to R7. The displayed registers contents of R0-R7 is with respect to the current bank selected. The information as to which bank is the current bank can be found by examining the bits RS1,RS0 of the flag register. Similarly a different bank can be selected by changing the bits RS1 and RS0 of the flag register.

RS1	RS0	Register Bank	Address
0	0	0	00H-07H
0	1	1	08H-0FH
1	0	2	10H-17H
1	1	3	18H-1FH

Key Pressed	Display			Comments
	Special Field	Address Field	Data Field	
RESET		-ESA	51	System Reset
EXAM REG		rE		Examine Register Command
A		A	00.	contents of Reg A
8		A	08.	New Data
NEXT		b	00.	A register updated Next Register's contents displayed.
EXEC		-		Command termination prompt.

3.4.3 GO COMMAND

FUNCTION

The GO command is used to transfer control of the system from the monitor to user's program.

FORMAT

GO [addr] EXEC.

OPERATION

1. To use this command, press the GO key when prompted for command entry. When this key is pressed, the current contents of the User Program Counter are displayed in the address field and

the contents of the location addressed by the PC are displayed in the data field. A dot also appears at the right edge of the address field indicating that the user can update the value of user Program Counter if required.

2. To begin program execution, press EXEC key. Now the address and data fields are cleared, and "E" is displayed at the left edge of the address field and control is then transferred to the user program at the current value of the user program counter.

3. To abort execution of user program press RESET key. However, in this case, all register information about user program is lost. Note that in any case the contents of the user portion of the RAM area are not disturbed by the Monitor.

4. There are two methods to break the user program execution.
   a) Set breakpoints at specific addresses and enable them.
   b) Press "BREAK" key

   When any of the specified enabled breakpoints is encountered, control returns to the monitor which saves all the registers and displays the address where the program broke and the data at that address on the display. It also displays the memory type in the special field. It will display"P" if the program broke in the program memory, or "d" if the program broke in the data memory. (Refer to section 3.4.4. for more details)

If "BREAK" key is pressed, control returns to the monitor which saves all the registers and displays the address where the program broke and the data at that address on the display. It will display "U" on the special field indicating that the program broke due to pressing of user break key.

Examples

Example 1 : Suppose the following program has been entered in the program memory by EXAM MEM command.

Location	Object Code	Mnemonic
8800	74	MOV A, # 42H
8801	42	
8802	F9	MOV R1, A
8803	80	SJMP 8800H
8804	FB	

To run this program, press the keys according to the following sequence.

Key Pressed	Display		Comments
	Address Field	Data Field	
RESET	-ESA	51	System Reset
GO	0000.	02	GO Command current PC and the byte at this PC are displayed
8	0008.		New Starting address
8	0088.		
0	0880.		
0	8800.		
EXEC	E		Control is transfered to the program at 8800H. The Execution is in progress.

Example 2: Example to show the use of "BREAK" Key.

Since the program in Ex 1 is in infinite loop user has to press RESET key to come out of the program, but now the register contents would be lost. Instead if you press "BREAK" key the program breaks immediately with the latest status of all the registers saved.

3.4.4. BREAKPOINT COMMANDS

INTRODUCTION

The ESA31 enables you to control program execution by setting breakpoints. A breakpoints is an address that stops program execution each time the address is encountered. By setting breakpoints at key addresses in your program, you can "freeze" program execution and examine the status of memory or registers at those points.

FUNCTION

These commands are used to set breakpoint, clear breakpoint, and display breakpoint, in both program memory and data memory, enable and disable breakpoint in both memory independently. The breakpoint can be one or more and also user can specify range of address for breakpoint.

3.4.4.1. CLEAR BREAK

FUNCTION :

This command is used to clear breakpoint in program memory or data memory.

FORMAT

CLRBRK {PRG MEM | EXT DATA} addr [[[next]...] EXEC]
[PREV <range-end addr>] EXEC

OPERATION:

1. To use this command, press CLR BRK key when prompted for command entry. A dot appears at special field prompting the user to specify the type of memory.
2. Press PRG MEM or EXT DATA key to select program memory or data memory.
3. A dot appears at the address field for user to enter breakpoint address. Enter the required breakpoint address.
4. User can enter one or more breakpoint address by pressing NEXT key and address and terminate the command with EXEC key. User can also specify a range of address to be cleared by pressing PREV key after entering the first address. Two dashes appear at the data field to indicate range. Enter the range's last address and press EXEC key to terminate or NEXT key to continue.

Example Following procedure clears breakpoint in program memory for range 0 to FFFFH.

Key Pressed	**Display**			**Comments**
	Special Field	**Address Field**	**Data Field**	
RESET		-ESA	51	System Reset
CLR BRK	-			Enter the comamnd
P	P		.	Enter the type of memory
0	P	0000.		Enter the start address of range
PREV	P	.	—	
F	P	000F.	—	Enter the last
F	P	00FF.	—	Address of range
F	P	0FFF.	—	
F	P	FFFF.	—	
EXEC		-		Terminate the command.

3.4.4.2. SET BREAK

FUNCTION

This command is used to set breakpoint in program memory or data memory.

FORMAT

SETBRK {PRG MEM | EXT DATA}addr [[[NEXT]...] EXEC]
[PREV <range -end addresss>] EXEC

OPERATION

1. To use this command, press SETBRK key when prompted for command entry. A dot appears at special field prompting the user to specify the type of memory.
2. Press PRG MEM or EXT DATA key to select program memory or data memory.
3. A dot appears at the address field for user to enter break point address. Enter the required breakpoint address.
4. User can enter one or more breakpoint address by pressing NEXT key and address and terminate the command with EXEC key. User can also specify a range of address to be set by pressing PREV key after entering the first address. Two dashes appears at the data field to indicate range. Enter the last range address and press EXEC key.

Example : Following procedure sets breakpoints in data memory for range 0 to 00FFH and 1000H

Key Pressed	Display			Comments
	Special Field	Address Field	Data Field	
RESET		-ESA	51	System Reset
SET BRK	.			Enter the Command
EXT DATA	d			Enter the type of memory
0	d	0000.	.	Enter the starting
PREV	d	.	---	address of range
F	d	000F.	---	Enter the last
F	d	00FF		address of range.
NEXT	d	.		
1	d	0001		
0	d	0010.		
0	d	0100.		
0	d	1000.		
EXEC		-		Terminate command

3.4.4. 3. DISPLAY BREAK

FUNCTION

This command is used to display breakpoint in program memory or data memory.

FORMAT

DP BRK { PRG MEM | EXT DATA} NEXT | EXEC

OPERATION

1. To use this command press DP BRK key when prompted for command entry. A dot appears at special field prompting the user to specify the type of memory.
2. Press PRG MEM or EXT DATA key to select program memory or data memory.
3. The first address is displayed in the address field. Two dashes appear in the data field and dot in the address field, if the breakpoint is a range of addresses with starting address displayed in the address field .
4. Press NEXT key for the next breakpoint address or last address of the range.
5. Terminate the command with EXEC key, or the command gets terminated if no more breakpoinnts are found

Example : Following procedure displays breakpoints in data memory

Key Pressed	**Display**			**Comments**
	Special Field	**Address Field**	**Data Field**	
RESET	-ESA		51	System Reset
DPBRK	.			Enter the command Enter the type of memory
EXT DATA	d	0000.	---	Starting address of range
NEXT	d	00FF	---	Ending address of the range
NEXT	d	1000		
EXEC				Terminate the command

3.4.4.4 DISABLE BREAK

FUNCTION

This command is used to disable the breakpoints which are previously set.

FORMAT

DABLE BRK {PRG MEM | EXT DATA} [NEXT {PRG MEM | EXT DATA}] EXEC

OPERATION

1. To use this command, press DABLE BRK key when prompted for command entry
2. A dot appears in the special field prompting the user to enter the type of memory
3. Press PRG MEM or EXT DATA key to select program memory or data memory
4. Press EXEC key.

Example: Following procedure disables the program and data memory breakpoints.

Key Pressed	Display			Comments
	Special Field	Address Field	Data Field	
RESET		- ESA	51	System Reset
DABLE BRK	.			Enter the type of
P	P			memory. Program
NEXT	.			memory and data
D	d			memory.
EXEC		-		Terminate the command

3.4.4.5 ENABLE BREAK

FUNCTION

This command is used to enable the breakpoint after setting the breakpoint.

FORMAT

EABLE BRK {PRG MEM | EXT DATA} [NEXT {PRG MEM | EXT DATA}] EXEC

OPERATION

1. To use this command, press EABLE BRK key when prompted for command entry.
2. A dot appears in the special field prompting the user to enter the type of memory.
3. Press PRG MEM or EXT DATA key to select program memory or data memory.
4. Press EXEC key.

Example : Following procedure enables program memory breakpoints

Key Pressed	Display			Comments
	Special Field	Address Field	Data Field	
RESET		-ESA	51	System Reset
EABLE BRK	.			Enable the
P	P			program memory
EXEC		-		Terminate the command.

3.4.5 SINGLE STEP COMMAND

FUNCTION

This command is used to execute a program, one instruction at a time. With each instruction executed, control is returned to the monitor. There is also an option to single step a program with count. Count determines the number of instructions to be executed at a time. Thus this command is an extremely useful debugging tool.

FORMAT

SINGLE STEP [start addr][NEXT count] EXEC [NEXT ...] EXEC

OPERATION

1. To use this command, press the SINGLE STEP key when prompted for command. Now the contents of the user program counter are displayed in the address field and the byte at this location is displayed in the data field.
2. To execute the one instruction at the current value of the program counter, press the EXEC key. When this key is pressed, the instruction at the current PC value is executed the new value of user PC is displayed in the address field and its associated instruction byte is displayed in the data field. Press NEXT key to continue single stepping.
3. To execute one instruction at the desired address, enter the address and now press the EXEC key. One instruction gets executed. Press NEXT key to continue single stepping.
4. To terminate command, press the EXEC Key. (Note that after terminating the SINGLE STEP Command if you again press the SINGLE STEP key, control returns exactly to the point where you pressed the EXEC key).
5. To single step a program with count, first press SINGLE STEP key. Now enter the desired starting address of the program and press NEXT key. Now a dot appears at the right edge of the data field prompting the user to enter the count. Enter the count (00 to FF) and press EXEC key. Now as many instructions as specified by the count get executed and the user can either continue stepping with count by pressing NEXT key or can terminate the command by pressing EXEC key.

EXAMPLES

Example 1. Suppose the program given as Example 1 to illustrate the GO Command has been entered in the memory. Now this program can be single-stepped as follows.

Key Pressed	**Display**			**comments**
	Special Field	**Address Field**	**Data Field**	
RESET		-ESA	51	System Reset
SINGLE STEP		XXXX.	XX	Current value of PC and the byte at this PC
8		0008.		
8		0088.		
0		0880.		
0		8800.		
EXEC		8802.	4F	Instruction at 8000 is executed Next instruction to be executed is displayed.
NEXT		8803	dF	Next instruction
EXEC		-		Command termination

NOTE : Single step command disables the breakpoint in program memory and data memory.

3.4.6 BLOCK MOVE COMMAND

FUNCTION

This command is used to move a block of data from one area of the memory to another area.

FORMAT

BLKMOVE {PRGMEM | EXT DATA | INT DATA}Start addr NEXT end addr
NEXT { PRGMEM | EXT DATA | INT DATA} dest. addr EXEC

OPERATION

1. To use this command, press the BLK MOVE key when prompted for command entry. When this key is pressed, display field is cleared and prompts the user to enter type of memory.
2. A decimal point appears at the right edge of the special field. Press PRG MEM, EXT DATA or INT DATA key to select program memory, external data memory or internal data memory.
3. Now enter the starting address of the block of data to be moved. Press the NEXT key. Now the display field is cleared again and two decimal points appear at the right edge of the address field. Now enter the ending address of the block of data to be moved. Press the NEXT key. Monitor prompts the user to enter destination type of memory by displaying a dot at the special field. Enter the destination type of memory.
4. A decimal point appears at the right edge of the address field. Now enter the starting address of the area into which the block of data is to be moved. This address is called the destination address. Press the EXEC key to start the command execution.
5. Monitor now moves the block of data from Start Address to End Address, to the area beginning at the Destination Address. After moving the data monitor displays the command prompt.

ERROR CONDITIONS

1. Specifying a value for the end address of the source block which is less than the value of the start address of the source block.
2. Trying to move the data into non-existent or Read Only Memory.

Example 1: Moving the block of program code listed in the example for the GO command to external data memory

Key Pressed	Display			Comments
	Special Field	Address Field	Data Field	
RESET		- E S A	51	System Reset
BLK MOVE	.			Block move command
P	P	. . .		type of memory

Key Pressed	Display			Comments
	Special Field	Address Field	Data Field	
8	P	0 0.0.8.		Start address
8	P	0 0.8.8.		
0	P	0 8.8.0.		
0	P	8 8.8.0.		
NEXT	P	. .		
8	P	0 0 0.8		End address source block
8	P	0 0.8.8.		
0	P	0 8.8.0.		
3	P	8 8 0.3		
NEXT	.			destination type of memory
EXT DATA	d	.		
8	d	0 8.8.0.		Destination address
8	d	0 0 8 8.		
2	d	0 8.8.2.		
0	d	8 8.2.0.		
EXEC		-		Block moved command prompt.

Now using the EXAM MEM key observe the contents of the locations of data memory 8820H, 8821H and 8822H, 8823H and verify that the code has indeed been moved into the destination block.

NOTE: BLOCK MOVE moves a block from one area to another area. It does not consider whether the block being moved consists of program code or data. Thus no relocations of program code occurs and the block is simply moved.

The system determines if the destination block overlaps the source block. It moves the data from either the starting address or from the ending address as required when overlap exists.

3.4.7. COMPARE COMMAND

FUNCTION

This command is used to compare two blocks of memory.

FORMAT

COMP {PRG MEM | EXT DATA | INT DATA} Start addr NEXT end addr NEXT
{PRG MEM | EXT DATA | INT DATA} dest addr EXEC

1. To use this command, press the COMP key when prompted for command entry. when this key is pressed, display field is cleared and prompts the user to enter type of memory.
2. A decimal point appears at the special field. press the PRG MEM, EXT DATA or INT DATA key to select program memory, external data memory or internal data memory.

3. Now enter the starting address of the block of data to be compared. Press the NEXT key. Now the display field is cleared again and two decimal points appears at the right edge of the address field. Now enter the ending address of the block of data to be moved. Press the NEXT key. Monitor clears the display field and prompt the user to enter destination type of memory by displaying a dot at the left most display. Enter the destination type of memory.
4. A decimal point appears at the right edge of the address field. Now enter the starting address of the area into which the block of data is to be moved. This address is called the destination address. press the EXEC key to start the command execution.
5. The monitor now compares the data in the specified source block against the data in the destination block. If no mismatch is detected in the entire address range specified, the command is terminated and the command prompt is displayed. If any mismatch is detected, the source address and data are displayed and the system waits for user's input. If user presses EXEC key, the command is terminated and the command prompt is displayed. If user presses the NEXT key, the system proceeds with the comparison from the next location.

ERROR CONDITION

1. Specifying a value for the end address of the source block which is less than the value of the start address of the source block.

EXAMPLES:

Example 1: Using Examine Memory command, enter the following data into memory locations 8800H to 8807 H.

8800 : 00, 01, 02, 03, 04, 05, 06, 07

Now use the block Move command to move this data into the destination block starting 8810H. Now compare these two blocks of memory.

Key Pressed	Display			Comments
	Special Field	Address Field	Data Field	
RESET		- E S A	51	System Reset
COMP	.			Compare command prompt for
P	P	. . .		source start
8	P	0 0.0.8.		type of memory
8	P	0 0.8.8.		
0	P	0 8.8.0.		
0	P	8 8.8.0.		
NEXT	P	. .		Prompt for source end.
8	P	0 0 0.8		
8	P	0 0.8.8.		

Key Pressed	Display			Comments
	Special Field	Address Field	Data Field	
0	P	0 8.8.0.		
7	P	8 8 0.7.		
NEXT	.			prompt for destn
	P			type of memory
8	p	0 0 0 8.		Destination address
8	p	0 0 8 8.		
1	p	0 8 8 1.		
0	p	8 8 1 0.		
EXEC		-		Match. command termination / command prompt.

Now using Examine Memory command, alter the contents of location 8805H and issue the compare command again. The key sequence will be exactly same as described already. Hence only the interaction after pressing EXEC key is shown.

EXEC	8805	55	Mismatch. waiting for user input
NEXT			continue comparison command termination / command prompt.

3.4.8 FILL COMMAND

FUNCTION

This command is used to fill a block of memory with a constant data

FORMAT

FILL {PRG MEM | EXT DATA | INT DATA} Start addr NEXT end addr
NEXT data EXEC

OPERATION

1. To use this command, press the FILL key when prompted for command entry. When this key is pressed, display field is cleared and decimal point appears at special field prompting the user to input the type of memory. Enter the type of memory.

2. Two decimal points appear at the right edge of the address field. Now enter the starting address of the block of memory to be filled. Press the NEXT key. Now the display field is again cleared and again a decimal point appears at the right edge of the address field. Now enter the ending address

of the block. Press the NEXT key. Monitor displays a decimal point at the right edge of the address field. Now enter the constant. Press the EXEC key to start the command execution.

3. Monitor now fills the block of memory from start address to end address with the speficied constant. Then the monitor displays the command prompt sign.

ERROR CONDITIONS

1. Specifying a value for the end address of the source block which is less than the value of the start address of the source block.
2. Trying to fill data in non-existent or ROM area.

Example:

Filling the block of program memory from 8800H to 880FH with a constant 55H.

Key Pressed	**Display**			**Comments**
	Special Field	**Address Field**	**Data Field**	
RESET		- E S A	51	System Reset
FILL				Fill command
P	P	. .		Type of memory
8	P	0 0 0.8 .		Start address
8	P	0 0 8.8.		
0	P	0 8 8.0.		
0	P	8 8 0.0.		
NEXT	P	.		
8	P	0 0 0 8.		End address of block
8	P	0 0 8 8.		
0	P	0 8 8 0.		
F	P	8 8 0 F.		
NEXT	P	8 8 0 F	.	
5	P	8 8 0 F	0 5.	Constant value
5	P	8 8 0 F	5 5.	
EXEC		-		Block filled. Command prompt.

Now using the EXAM MEM key observe the contents of the locations 8800H to 880FH and observe the data to be 55H.

3.4.9 SEARCH COMMAND

FUNCTION

This command is used to search a block of memory for a string of data upto 4 values.

FORMAT

SERCH {PRG MEM | EXT DATA | INT DATA} start addr NEXT end addr
NEXT data1 [[[NEXT data2] NEXT data3] NEXT data4] EXEC

OPERATION

1. To use this command, press the SERCH key when prompted for command entry. When this key is pressed, display field is cleared and decimal point appears at special field prompting the user to input the type of memory. Enter the type of memory.
2. Two decimal points appear at the right edge of the address field. Now enter the starting address of the block of memory to be searched. Press the NEXT key. Now the display field is cleared again and a decimal point appears at the right edge of the address field. Now enter the ending address of the block. Press the NEXT key. Monitor clears the display field and a decimal point appears at the right edge of the data field.

   Now enter the constant to be searched upto 4 data. Press the EXEC key to start the command execution.
3. Monitor now searches block of memory, from Start Address to End Address for the specified string. Then the monitor displays the address of the first location at which the string is found.
4. Press NEXT key to continue the search or EXEC key to terminate the command.

ERROR CONDITION

1. Specifying a value for the end address of the source block which is less than the value of the start address of the source block

Example : Searching the block of program memory from 8800H to 880FH for a constant 55.

Key Pressed	**Display**			**comments**
	Special Field	**Address Field**	**Data Field**	
RESET		-ESA	51	System Reset
SEARCH	.			Search command
P	P	. .		Type of memory
8	P	0008.		Start address
8	P	0088.		
0	P	0880.		

0	P	8800.		
NEXT		.		
8	P	0008.		End address of block
8	P	0088.		
0	P	0880.		
F	P	880F.		
NEXT			.	
5			05.	Constant value
5			55.	
EXEC	-			Start search
	P	8800	55	
EXEC		-		Command termination prompt

CHAPTER 4

SERIAL MONITOR

4.1. INTRODUCTION

This chapter describes the commands supported by the Serial Monitor Program. The Serial Monitor allows ESA31 to be operated from a CRT terminal/Host Computer connected via the RS-232-C serial interface. (refer to chapter 5 on Hardware and Appendix C on RS-232-C connector Details).

The system must be configured for serial mode of operation as described in section 2.1.1. The commands are described in this chapter.

When the system enters serial mode of operation, the sign-on message “ESA-8051 Serial Monitor V x. y” is displayed (x is the current version number and y is the revision number) on one line and a prompt “>” on the next line indicating that the monitor is ready to accept commands from the user. With the exception of RESET key, and BREAK key the keyboard is completely disabled.

4.2 STRUCTURE OF MONITOR COMMANDS

Whenever the monitor is ready to accept a command from the user, it outputs a greater than symbol (‘>’) as the command prompt character at the beginning of a new line.

The commands entered by the user consists of a single character command mnemonic followed by a list of command parameters. This list may consist of upto four parameters depending on the particular command being used. When more than one parameter is required, a single (‘,’) or space is used between the parameters as a separator.

A command is terminated by a Carriage Return. Commands are executed one at a time and only one command is allowed within one command line.

PARAMETER ENTRY

All numeric parameters are to be entered as hexadecimal numbers. The valid range for one byte parameters is 00 to FF and if more than 2 digits are entered, only the last two digits are valid (leading zeros may be omitted). Thus all one byte values are interpreted modulo 256 (decimal). The valid range for 2-byte parameters is 0000 to FFFF and longer values are evaluated modulo 64K (i.e. only the last four digits are valid).

All the commands except the R (examine/modify register) command require only hexadecimal values as parameters. The register name abbreviation entries required by the R command are described later while describing the R command in detail.

RESPONSE TO ERRORS

Whenever an error is detected by the monitor (either in the command entry or in the command execution) the command is aborted, “Error” is displayed on the next line with a “^” sign attached pointing to the place where the error occured and a new command prompt is issued. (The possible error conditions are described while illustrating the individual commands.)

Command execution occurs only after a valid delimiter (a carriage return)is entered. Hence a command entry can be canceled anytime before the delimiter is entered by pressing “ESC” key.The command prompt character is output on a new line.

4.3 MONITOR COMMANDS

Each command described in this chapter consists of a one or two character, followed by appropriate parameters and data. These commands are summarized in Table 4.1. and are described in detail in the sections which follow. In the table as well as in the subsequent descriptions, the following notation is used:

NOTATIONAL CONVENTIONS

SYMBOL	NAME	USAGE
{}	curly braces with ellipsis	Encloses a required argument 1 or more times.
[]	square brackets	encloses an item that appears 0 or 1 time.
[]...	square brackets	encloses an item that appears 0 or more times
|	vertical bar	seperates alternative items in a list.
	italics	indicates a descriptive item that should be replaced with an actual item.

TABLE 4.1 SUMMARY OF SERIAL MONITOR COMMANDS

COMMAND	FUNCTION/FORMAT
A	Assembler A [address]
B	Clear/Display/Set/ breakpoint in program memory, external data memory B{[D{P|D}]|[{C|S} {P|D} address1 [,address2]]}
C	Compare a block of memory with destination block. C{P|D|I {address1,address2,{P|D|I}, address3
D	Disable breakpoint in program memory or data memory. D{P|D}
E	Enable breakpoint in program memory or data memory. E{P|D}
F	Fill a block of memory with a constant or search a string of data in program memory, external data memory and internal data memory . F {P|D|I}, address1,address2, data [,data[,data [,data]]],S
G	Transfers the processor control from the monitor to user program breakpoints. G [address]
H	Help. List all the commands supported by the Serial Monitor H
J	Jump to address J address
L	Load data from tape(Refer Chapter 7)
M	Modify/Display/Move memory contents in program memory, external data memory and internal data memory with all combinations M {P|D|I|B} address1 [,address2 [,{P|D|I},address3]]
N	Execute one or more instructions specified by user. N {count}
P	Programmer

S	Execute one Instruction of user program S[R][address]
W	Write data onto tape (Refer chapter 7)
Z	Disassembler Z[addr1[,addr2]]
[U & V]	commands are reserved for system usage.

4.3.1 M (MODIFY MEMORY) COMMAND

FUNCTION

The M (Modify Memory) command is used to examine the contents of specified memory locations. Further, if the locations are in RAM their contents can be altered if desired and block move contents of memory from program, data, internal memory to program, data, internal memory for all combinations.

FORMAT

M {P|D|I|B} address1 [,address2 [,{P|D|I}, address3]]

OPERATION

1. Enter M followed by memory type, the address of the memory location to be examined and then enter <CR>. The monitor will now output the contents of that location.Note that in serial mode a '-' is always a prompt for data entry, while a ">"is the prompt for command entry.
2. To modify the contents of this location, the user can enter the new value now.
3. Enter a Carriage Return, either immediately after the '-' prompt by the system or after the entry of a new value, to examine/modify the next sequential location. A "<ESC>"instead of the <CR> terminates the command and returns the monitor to the command entry mode.

ERROR CONDITION:

1. Trying to modify the contents of non-existent or PROM locations.

Example 1: Examine the PROM locations 11H

>MP11 <CR>
>0011 FF <ESC>
>

Example 2: Examine a series of RAM locations starting at 8820H and modify the contents of the location 8822H.

>MD8820<CR>
8820 xx <CR>
8821 xx <CR>
8822 xx 55 <CR>
>8823 xx <ESC>

4.3.2 M (DISPLAY MEMORY) COMMAND

FUNCTION

This command is used to display the contents of the program or external or internal data memory.

FORMAT

M {P|D|I}, address1, address2 <CR>

OPERATION

1. To use this command, enter M when prompted for command entry, After entering M, enter the memory type and then starting address of the memory block whose contents are to be displayed, then enter a comma, enter the end address of the memory block and follow it with a Carriage Return.
2. Now the monitor will output the starting address, the contents of the location from this address to the specified end address. The display appears in formatted lines with 16 bytes/line. The number of bytes displayed on the first line are so adjusted that if the second line is present, its first location has address with the last nibble as zero. The ASCII equivalents of the displayed data values are also shown on each line. The non- displayable characters are shown as periods (“.”).

Examples

Example 1: To display the contents of 5 bytes from location 8000H.

>MD8000,8004

```
      0   1   2   3   4   5   6   7   8   9   A   B   C   D   E   F    ASCII
8000:41   42  43  0D  31                                              ABC.1
```

4.3.3 M (MOVE MEMORY) COMMAND

FUNCTION

This command is used to move a block data from one area of the memory to another area.

FORMAT

M {P|D|I}, <address1>, <address2>,{P|D|I}
<destination address> <CR>

OPERATION

1. To use this command, enter M when prompted for command entry. Follow it with the type of memory, starting address of the source block to be moved (“start address”), a comma, the ending address of the source block (“address2”), another comma, and then type of destination memory, starting address of the area into which the source block is to be moved (“destination address”). Now enter the carriage return.
2. This operation moves the contents of memory locations from “address1” to “address2” to consecutive memory locations starting from the “destination address”.
3. The system determines if there is any overlap between source and destination blocks and accordingly transfers the data beginning either at the “address1” or at the”address2".

ERROR CONDITIONS:

1. Specifying an “end address” value which is less than the value of the “start address”.
2. Trying to move data into non-existent or Read-Only Memory locations.

Examples:

Example 1. Move the contents of the locations 800H through 80FH to the memory block beginning at 8840H.

>MP800, 80F,P8840<CR>

Example 2

>MP800, 80F,P200<CR>

error

An attempt to move data into PROM locations produces the error message.

4.3.4 F (FILL MEMORY) COMMAND

FUNCTION

This command is used to fill a block of memory with a specified constant.

FORMAT

F {P|D|I}, address1 ,address2, constant <CR>

OPERATION

1. To use this command enter F when prompted for command entry and enter the type of memory to be filled.
2. Now enter the starting address of the block of memory to be filled. Enter a comma. Now enter the ending address of the block. Again enter a comma. Now enter the constant. Press the Carriage Return key to start the command execution.
3. Monitor now fills the block of memory from address1 to address2 with the specified constant. Then the monitor displays the command prompt sign.

ERROR CONDITIONS

1. Specifying a value for the address2 of the source block which is less than the value of the address1 of the source block.
2. Trying to fill the data in non-existent or read-only memory.

EXAMPLES

Example1: Filling the block of program memory with a constant 55H.

>FP8800, 880F,55 <CR>

Now you can use the M command to examine the block of memory to see that it is filled with the constant 55H.

4.3.5 C (COMPARE MEMORY) COMMAND

FUNCTION

Compare command can be used to compare the contents of one memory block with the contents of another memory block.

FORMAT

C {P|D|I}address1 of block 1,address2 of block1
{P|D|I} address1 of block 2 <CR>

OPERATION

1. To use this command, enter C when prompted for command entry and select the type memory.Then enter starting address of the first block, a comma, ending address of the first block, another comma and destination type of memory and comma and then the starting address of the second block followed by the Carriage Return.
2. The monitor now compares the contents of location beginning at address1 of block1 with the contents of location beginning at address1 of block2. This process continues till the contents of address2 are compared with those of the corresponding location in the 2nd block. Any differences detected are displayed.

Examples:

1. Compare the contents of memory locations 8000H to 8FFFH with those of a memory block beginning at 9000H

>CP8000, 8FFF,P9000 <CR>
>

(This response showed that there is no mismatch)

2. CPA000, AFFF,P8000 <CR>

ABC0 = 00 FF = 8BCO
AED8 = 48 54 = 8ED8

(This response showed that there is mismatch at two locations).

4.3.6 R (EXAMINE/MODIFY REGISTERS) COMMAND

FUNCTION

This command is used to examine and optionally modify the contents of the registers.

FORMAT

R [reg] [[[new data] ,]...] <CR>

OPERATION

1. To examine the contents of all the registers, enter R followed by carriage return when prompted for command entry. The monitor will now display the contents of all the registers.

2. If you wish to examine/modify the contents of a particular register, then enter R (when prompted for command) followed by the register name abbreviation. The register name abbreviations are shown in Table 4.2. Now the monitor will output an equal sign ('='), the current contents of the specified register and data prompt character ("-"). The contents of this register can be changed now by entering the new data value, followed by a valid terminator (a Carriage Return or escape). If the terminator is the escape, the command is terminated.

   If the terminator is not the escape the next "sequential" register is displayed and opened for optional modification. The sequence in which registers are displayed is also shown in Table 4.2. (Note that this sequence is circular).

TABLE 4.2

Register name	Abbreviation
Register A	A
Register B	B
Stack Pointer	SP
Flags Register	PSW
Data Pointer High	DPH
Data Pointer Low	DPL
Register TH0	TH0
Register TL0	TL0
Register TH1	TH1
Register TL1	TL1
Register P1	P1
Register P3	P3
Program Counter High	PCH
Program Counter Low	PCL
Register R0	R0
Register R1	R1
Register R2	R2
Register R3	R3
Register R4	R4
Register R5	R5
Register R6	R6
Register R7	R7

NOTE: The flags register PSW is also displayed in bit format when Register command is executed.The meaning of the pattern "CAFBBOGP" is as follows.

PSW bit	Abbreviated as	Function
PSW.7	C	Carry Flag
PSW.6	A	Auxiliary Flag
PSW.5	F	Flag 0 available to the user
PSW.4	B	Register Bank selector bit 1
PSW.3	B	Register Bank selector bit 0
PSW.2	O	Overflow Flag
PSW.1	G	usable as a general purpose flag
PSW.0	P	Parity Flag

Examples :

Example 1

> R <CR>

A	B	SP	PSW	DPH	DPL	TH0	TL0	TH1	TL1	P1	P3	PCH	PCL
(E0)	(F0)	(81)	(D0)	(83)	(82)	(8C)	(8A)	(8D)	(8B)	(90)	(B0)		
00	00	07	00	00	00	00	00	00	00	FF	FF	00	00

R0	R1	R2	R3	R4	R5	R6	R7	PSW
(00)	(01)	(02)	(03)	(04)	(05)	(06)	(07)	C A F B B O G P
00	00	00	00	00	00	00	00	0 0 0 0 0 0 0 0

>

Example 2

Examine and alter register A and then examine register B.

RA <CR>

A (E0) 24 <CR>
B (F0) 25 <ESC>

4.3.7 J (JUMP TO ADDRESS -SET/CHANGE PC) COMMAND

FUNCTION

The J command is used to change the Program Counter value to the desired address before executing a program by either GO command or SINGLE STEP command.

FORMAT

J[address] <CR>

OPERATION

To use this command, enter J when prompted for command entry. Now enter the desired starting address of the program you wish to execute. Enter Carriage Return. Now command prompt reappears on the next line.

4.3.8 G (GO) COMMAND

FUNCTION

The GO command is used to transfer the control of the system from monitor to the user's program.

FORMAT

G[address1] <CR>

OPERATION

To use this command, enter G when prompted for command entry. Execution starts from the PC value.

Now, if you wish to modify the value of the PC (i.e. the address to which control is to be transferred), enter the new value. Enter Carriage Return. Now the user context is restored and control is transferred to the program starting at the current value of the user program counter.

A powerful debugging tool breakpointing a program is available to the user. To use this facility, set one or more breakpoint in program or data memory using B command.

Now the control is transferred to the program starting at the current PC value. Upon reaching any one of the specified breakpoint addresses control is returned to the monitor. Monitor saves the complete user context, displays the current PC value and then issues a command prompt.

Notes:

1. When any one of the breakpoints is reached, control is returned to the monitor, after saving the registers.
2. Specifying more than one breakpoint addresses is useful when debugging a program section containing branch instructions.

EXAMPLE

Enter the program presented as example 1 for the GO command from the keyboard monitor (section 3.4.5). You can execute it as shown below:

>G 8800 <CR>

The program can be executed by setting up breakpoints.

The procedure is as follows.

1. Set a breakpoint in the program memory or data memory using

BSP address <CR>

2. If the desired breakpoints is a range, then enter BSP addr1, addr2 <CR>
3. Enable breakpoint using EP command or ED command for program or data memory respectively.
4. Execute the program using G command. (Note: single step execution of program memory disables breakpoint memory).
5. Cause of break with the program break address are displayed.
6. Enter Carriage Return key to continue.The program starts executing from the point at which break has occurred.
7. The above procedure is repeated if program encounters another breakpoint.
8. Enter 'ESC' key to terminate the process.

4.3.9 S (SINGLE STEP) COMMANDS

The ESA31 Trainer enables you to debug a program by single stepping the instructions. The command is used to execute a program one instruction at a time. With each instruction executed,control is returned to the monitor. Thus this command is an extremely useful debugging tool. Provision has been made for single stepping with register display, disassembly and count.

4.3.9.1 SR (SINGLE STEP WITH REGISTER DISPLAY) COMMAND

FUNCTION

This command is used to single step a program with register display.

FORMAT

SR [addr] <CR>

OPERATION

1. To use this command, enter SR when prompted for command.
2. To execute one instruction at the current value of the program counter, press the "CR" key. When this key is pressed, the instruction at the current PC value is executed and then all the register values are displayed.
3. To execute one instruction at the desired value of PC, enter SR and the desired starting address of the program and then press the "CR" key.
4. To execute instructions one by one there are two ways.
   a) Press 'CR' key whenever you want to execute one instruction at a time. In this case single stepping is totally under users control. Each time 'CR' key is pressed one instruction is executed. To terminate the command press 'ESC' key.
   b) Press space-bar whenever you want to continuously single step a program. Now single stepping will be continuous and when you want to stop and see registers press space-bar again, now single stepping is stopped. At this stage you can continue single stepping by pressing space-bar or execute one instruction at a time by pressing 'CR' key or can terminate command by pressing 'ESC' key.

EXAMPLE

Example 1. Suppose the program given as example 1 to illustrate the GO command has been entered in the memory. Now this program can be single-stepped as follows.

```
>SR 8800 <CR>

A     B     SP    PSW   DPH   DPL   TH0   TL0   TH1   TL1   P1    P3    PCH         PCL
(E0)  (F0)  (81)  (D0)  (83)  (82)  (8C)  (8A)  (8D)  (8B)  (90)  (B0)
00    00    07    01    00    00    00    00    00    00    FF    FB    88          02

R0    R1    R2    R3    R4    R5    R6    R7                PSW
(00)  (01)  (02)  (03)  (04)  (05)  (06)  (07)          CAFBBOGP
00    00    00    00    00    00    00    00          0 0 0 00 0 01
8802        F9          MOV R1,A
>
```

4.3.9.2 S (SINGLE STEP COMMAND WITH DISASSEMBLY)

FUNCTION

This command is used to single step a program with disassembly. The register contents will not be displayed.

FORMAT

S [addr] <CR>

OPERATION

1. To use this command enter S when prompted for command. Rest of the procedure is same as for the SR command. Only executed instructions in disassembled format are displayed and register contents are not displayed.

```
 >S 8800 <CR>
8802        MOV         R1,A
```

4.3.9.3 N (SINGLE STEP WITH COUNT)

FUNCTION

This command is used to single step a program with count. The maximum value of the count is FF. Multiple instructions can be executed at a time.

FORMAT

N (count) <CR>

OPERATION

1. To use this command, set the PC value to the starting address of the program using J command.

2. Enter N, the count and press “CR” key.

If the program starting address is 9000H and the count is 20 instructions at a time, following commands have to be executed

>J 9000 <CR>
>N 14 <CR>

Now 20 instructions will be executed at a time. The register contents are displayed just like in SR command. Now the user can either continue single stepping by pressing "CR" key or space bar or the user can exit from the command by pressing ‘ESC’ key.

4.3.10 B (BREAKPOINT) COMMANDS

BREAKPOINTS:

The ESA31 enables you to control program execution by setting break points. A breakpoint is an address that stops program execution each time the address is encountered. By setting breakpoints at key addresses in your program, you can “freeze” program execution and examine the status of memory or registers at that point.

These commands are used to set breakpoint,clear breakpoint and display breakpoint in both program memory and data memory, enable and disable breakpoint in both memory independently. The breakpoint can be one or more and also user can specify range of address for breakpoint.

4.3.10.1 CLEAR BREAKPOINT

FUNCTION

To clear the breakpoint(s) in the data memory or program memory.

FORMAT

BC{P|D}addr1[,addr2]

OPERATION

To clear the breakpoint enter BCP or BCD for corresponding memory and with one address or range of address and Carriage Return.

Example : To clear the breakpoint of full program memory, enter
>BCP 0,FFFF <CR>

After clearing procedure is finished a command prompt is displayed.

4.3.10.2 SET BREAKPOINT

FUNCTION

The set break command is used to set breakpoints in program memory and data memory.

FORMAT

BS{P|D}addr1[,addr2]

OPERATION

1. Set a breakpoint in the program memory or data memory using BSP <address> <CR>
2. If the desired breakpoints is a range, then enter BSP addr1, addr2 <CR>

Example

To set a breakpoint in the program memory at address 000BH enter the following command.

>BSP 000B <CR>

4.3.10.3 DISPLAY BREAKPOINT

FUNCTION

To display the breakpoint which has been set using BSP or BSD command, enter BDP or BDD for program memory or data memory respectively with carriage return.

FORMAT

BD{P|D} <CR>

OPERATION

1. Enter the command BDP or BDD and carriage return to display the preset breakpoints.
2. Enter Carriage Return to view remaining preset break point(s) or terminate the command with 'ESC' key.
3. ‘No breakpoints found ‘ message is displayed if no breakpoints are set.

Example

To display the breakpoint in the program memory which has been set in the previous command.

>BDP <CR>
000B <ESC>

The above address is displayed to indicate that breakpoint at address 000bH is set in the program memory.

4.3.10.4 ENABLE BREAKPOINT

FUNCTION

To enable the breakpoint which has been set using BSP or BSD command

FORMAT

E{P|D} <CR>

OPERATION

Enter the command EP or ED and carriage return to enable the preset breakpoint(s).

Example

To enable the breakpoint in the program memory which has been set in the previous command.

>EP <CR>

4.3.10.5 DISABLE BREAKPOINT

FUNCTION

To disable the breakpoint which has been set using BSP or BSD command

FORMAT

D{P|D} <CR>

OPERATION

Enter the command DP or DD and carriage return to disable the preset breakpoint(s).

Example

To disable the breakpoint in the data memory

>DD <CR>

>

Disable breakpoint does not clear the breakpoints

4.3.11 A (ASSEMBLY) COMMAND

ESA31 provides a powerful, PROM-resident Assembler to simplify the user's task of program development. This assembler, available in serial mode of operation, is a on- line one and supports all the standard mnemonics and addressing modes of Intel 8031 / 8051 microcontroller.

FUNCTION

The assembler generates the actual machine codes and stores them in the memory locations defined by the program. Also, the system will display the codes generated as well as the source statement. Any errors detected are also displayed on the screen.

OPERATION

'A' command implements the assembly facility. So, to invoke the assembler, type A with optional address when prompted for the command by the Serial Monitor.

>A [address] <CR>
Assembly language instructions consist of three fields, as shown below:

Address Object Mnemonic

The fields may be separated by any number of blanks and tabs but must be separated by at least one delimiter. Each instruction must be entered on a single line terminated by a Carriage Return. No continuation lines are possible.

Opcode Field

This required field contains the mnemonic operation code for the 8031 instruction to be performed.

Operand Field

The operand field identifies the data to be operated on by the specified opcode. Some instructions require no operands. Others require one, two or three operands. As a general rule, when two operands are required (as in data transfer and arithmetic operations), the first operand identifies the destination (or target) of the operation's result, and the second operand specifies the source data and the two operands must be separated by a comma.

Examples:

>A8000 <CR>

Address	Opcode	Mnemonics
8000	E9	MOV A,R1 <CR>
8001	74 42	MOV A,#42H <CR>
8003		<ESC>

>

4.3.12 Z (DISASSEMBLER) COMMAND

Disassembly is an extremely useful technique, often employed during debugging.

FUNCTION

A Disassembler converts machine language codes into assembly language mnemonics, making it easy for the user to understand/verify the program.

OPERATION

To use this facility, type Z when prompted for command by the Serial Monitor.

>Z [addr1[,addr2]] <CR>

Address Object Mnemonic

The disassembled code is displayed according to the above format.

The display can be halted at any point by Ctrl-S and restarted by Ctrl-Q. The disassembly can be aborted at any time by pressing Esc key.

NOTE: If the disassembly of the last instruction requires reading of data from locations beyond the specified address2, the system will read them to complete the disassembly. For example, if the specified Address2 is 81FFH and the code at 81FFH is 20H (which is a 3-byte instruction), the system will read the required data from locations 8200H and 8201H to complete the disassembly.

Example

>Z 0,6 <CR>

Address	Object	Mnemonic
0000	02 00 30	LJMP 0030H
0003	02 2A BE	LJMP 2ABEH
0006	FF	MOV R7,A

>

4.3.13 H (HELP) COMMAND

FUNCTION

The HELP command is used to list all the commands supported by the serial monitor.

FORMAT

>H <CR>

OPERATION

As soon as H and <CR> is entered by the user, in response to the command prompt, the system lists, in alphabetical order all the commands supported by the serial monitor. The display appears as shown below.

>H<CR>

Command	**Syntax**
Assemble	A [address]
Breakpoint	B {[D{P|D}]|[{C|S} {P|D} address1[,address2]]}
Compare Memory	C {P|D|I} address1,address2,{P|D|I},address3
Disable Brkpnt	D {P|D}
Enable Brkpnt	E {P|D}
Fill/Search MEM	F {P|D|I},address1,address2,data [,data[,data[,data]],S]
Go (Execution)	G [address]
Help	H
Jump	J address
Load from Tape	L filename
Memory	M {P|D|I|B}address1[,address2[,{P|D|I},address3]]
N (SS Count)	N count (Single Step Count = <0FFH>)
Register	R [register]
Single step	S [R] [address]
Write to tape	W {P|D} address1, address2, filename
Z - Disassembly	Z [address1 [,address2]]

U & V Commands are reserved for system usage.

CHAPTER 5

HARDWARE

5.1 INTRODUCTION

This chapter describes the hardware design details of ESA31. Appendix C gives the connector details and Appendix A has the component layout diagram. The design details are discussed in the following order:

a) CPU, Address Bus, Data Bus and Control Signals
b) Memory Addressing
c) Keyboard/Display Interface
d) Programmable Interval Timer and Serial Interface
e) Programmable Peripheral Interface Devices
f) Bus Expansion
g) Connector Details

5.2 CPU, ADDRESS,DATA AND CONTROL SIGNALS

ESA31 Uses 8031 / 8051 Microcontroller operated with a 11.0592MHz crystal. The on-board RESET key can provide a RESET signal to the CPU. The lower address bus is demultiplexed using a 74 LS 373 at U4 and the upper address bus is buffered using 74 LS 373 at U18. The data bus is buffered using a 74 LS 245 at U7.

5.3 MEMORY ADDRESSING

ESA31 has four 28-pin JEDEC compatible slots (U8,U9,U10 and U11) for accepting memory devices. The

socket at U8 is populated with a 27256 as program memory which contains the system firmware. The socket at U9 is populated with a 62256 to provide 32K bytes of static RAM of user program memory.

The socket at U10 & U11 is populated with a 62256 to provide 56K bytes of static RAM of user data memory.

The memory map is as follows:

TABLE 5.1 Memory Map

Device	Address range	Type of memory
27256 at U8	0000-7FFF	Program memory
62256 at U9	8000-FFFF	User Program memory
62256 at U10	0000-7FFF	User Data memory
62256 at U11	8000-DFFF	User Data memory

Battery Option:

The 62256 provided at U9, U10 and U11 can be backed up by an optional battery. The terminals for connecting the battery are brought out as BATT.

5.4 I/O ADDRESSING

I/O decoding is implemented using a E561,E562,E563 and E564.Thus foldback exists over the unused address lines. The I/O devices, their addresses and their usage is summarized below:

TABLE 5.2 I/O ADDRESS MAP

I/O Device	Address	Usage
8255A at U35		Available to user.
Port A	E900H	
Port B	E901H	
Port C	E902H	The signals are avai-
Control Port	E903H	able on connector J1
8255A at U27		Available to user
Port A	E800H	
Port B	E801H	The signals are avail-
Port C	E802H	able on connector J2
Control Port	E803H	
8253-5 at U13		Available to user
Timer 0	EA00H	Timer 0 is available to user on connector J3.

I/O Device	Address	Usage
Timer 1	EA01H	Timer 1 is available to user on connector J3.
Timer 2	EA02H	Timer 2 is available to user on connector J3.
Control Port	EA03H	
8251A at U14		Used for implementing serial Communication.
Data Port	EB00H	
Command port	EB01H	
8279-5 at U24		Used for implementing keyboard/display interface
Data Port	EC00H	
Command Port	EC01H	
8155: at U15		Reserved for system
Internal RAM	E000H-E0FFH	Reserved for system
Command/Status	E100H	
Port A	E101H	Parallel printer interface data
Port B	E102H	To read the DIP switch status
Port C	E103H	Printer interface and audio tape interface.

5.5 KEYBOARD/DISPLAY INTERFACE

The Keyboard/Display section of ESA31 is controlled by 8279. The port addresses are given in section 5.4. The 8279 is configured for the following operation mode:

* Encoded scan keyboard with 2-key lock out

* 8 digits, 8-bit, left entry display

The keyboard reading is implemented by polling the command/status port of 8279. The codes assigned to the keys of the on-board keypad are listed below:

TABLE 5.3

Serial No	Key	Corresponding Code
1	0/PRRD	00H
2	1/P1/BLNK	01H
3	2/PROG	02H
4	3/P3/VRFY	03H
5	4/SP	04H
6	5/PSW	05H

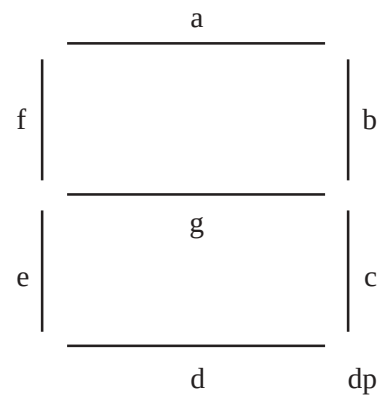
Serial No	Key	Corresponding Code
7	6/PCH	06H
8	7/PCL	07H
9	8/DPTH	08H
10	9/DPTL	09H
11	A/TPRD	0AH
12	B/TPWR	0BH
13	C/TH0	0CH
14	D/TL0	0DH
15	E/TH1	0EH
16	F/TL1	0FH
17	SET BRK	10H
18	CLR BRK	11H
19	EABLE BRK	12H
20	DABLE BRK	13H
21	NEXT/DPBRK	14H
22	PREV/SERCH	15H
23	PRG MEM	16H
24	EXT DATA	17H
25	FILL	18H
26	COMP	19H
27	SINGLE STEP	1AH
28	BLK MOVE	1BH
29	EXAM REG	1CH
30	EXAM MEM	1DH
31	GO	1EH
32	EXEC	1FH
33	BIT MEM	20H
34	INT DATA	21H

NOTE: RESET and BREAK keys are not connected to keyboard controller.

Display Drive:

The segment drive outputs of 8279 (A0 through A3 and B0 through B3) form a single 8-bit parallel output driving the display element segments. The correspondence between the data bus, segment outputs of 8279 and the display element segments is shown below:

TABLE 5.4



Display segments

Display Segment Control								
CPU DATA BUS	D7	D6	D5	D4	D3	D2	D1	D0
8279 OutPut	A3	A2	A1	A0	B3	B2	B1	B0
Segment enabled	d	c	b	a	dp	g	f	e

Bit = 1 Corresponding segment is ON
= 0 Corresponding segment is OFF

Example: To display “E”, the data format requires that segment a,f,g,e and d should be ON and other segments should be OFF. So the data should be 1001 0111=97H. Display codes for other patterns can be worked out similarly.

Display Position:

To display characters in the address field, 8279 must be instructed to start from display position 0 (by sending 90H to the command port of 8279) and to display characters in the data field, 8279 must be instructed to start from display position 4 (by sending 94H to the command port of 8279.).

5.6 PROGRAMMABLE INTERVAL TIMER

ESA31 has an on-board programmable interval timer 8253-5 at socket position U13. Its I/O addresses can be found in Table 5.2 in section 5.4.

8253 has one command/status port and three data ports called Timer0, Timer1 and Timer2 to provide three programmable timers. All the timers are available for user.The signals related to Timer 0,1 and 2 are available on system connector J5. Connector details are provided in the last section of this chapter.

5.7 SERIAL INTERFACE

An 8251A (Programmable communication interface) at position U14, is used for implementing RS-232- C compatible serial interface. This device is programmed for asynchronous operation, 2 stop bits, no parity, data length of 8 bits and baud scale factor of 16X. As already noted, Timer of 8155 is used to generate the required Transmit and Receive clocks of 8251A based on the setting of the DIP Switch.(Refer section 2.1.3 on Baud Rate selection).

The I/O address of 8251A can be found in Table 5.2 in section 5.4. The required voltage level conversion is

done by MAX 232 at U26. The RS-232-C compatible signals are available on 9-pin D type female connector J7. (Refer the last section of this chapter for connector details.)

NOTE:

If the handshake signals are not required, user can short DTR and DSR.

5.8 PROGRAMMABLE PERIPHERAL INTERFACE DEVICES

ESA31 has two numbers of 8255As (Programmable peripheral interface devices.). Each 8255A consists of a command port and three 8-bit programmable input/output ports called Port A, Port B and Port C. The Port addresses of these devices can be found in Table 5.2 in section 5.4.

The two 8255As at U35 and U27 are completely available to user. The port signals are available on connectors J1 and J2 (Refer the last section of this chapter for connector details.)

5.9 PROGRAMMABLE INPUT/OUTPUT AND TIMER

ESA31 has one 8155 (programmable I/O and timer).The 8155 has 256 bytes of RAM,two 8-bit and one 6-bit programmable input/output Port A, Port B, Port C, command port and two 8- bit registers to load counter for timer.

The Port A of the 8155 is configured as output Port to provide data for printer interface.The Port B is configured as input Port.The B0-B5 are used to read the DIP Switch of the trainer.The Port B bit B6, is used as busy signal for printer interface.The B7 bit inputs from an EAR socket for Audio Tape interface.

The Port C is configured as output port.The C0 bit gives strobe for printer interface.The bit C1 is used to send data for audio tape interface.

The Timer-in is fed with clock generated by crystal clock generator of 1.5MHz.The timer-out of the 8155 gives baud clock for serial communication with respect to DIP Switch setting.

VECTOR ADDRESS LOOK-UP TABLE FOR INTERRUPTS

Function	Interrupt Source	Vector Address	Trainer Address
External Interrupt 0	IE0	0003H	2CB0H
Timer Interrupt 0	TF0	000BH	FFF0H
External Interrupt 1	IE1	0013H	FFF3H
Timer Interrupt 1	TF1	001BH	FFF6H
Serial Interrupt	RI & TI	0023H	FFF9H

5.10 BUS EXPANSION

ESA31 permits easy expansion of the system by providing all the necessary signals on two connectors, J4 and J5. The signals are STD bus compatible and thus user can easily expand the capabilities of ESA31.

5.11 CONNECTOR DETAILS

There are six connectors on ESA31 in addition to the power connector J6. Of these, J1 and J2 are 26pin I/O connectors for 8255 ports, J5 is the system connector (providing address, data and control signals) and J4 is the user expansion (providing additional signals, for system expansion). J3 is the printer connector providing centronics parallel printer interface. J7 is the RS-232-C serial connector.

The signal definitions on all these connectors are listed below. (This information is available in Appendix B also)

J1 PORTS CONNECTOR

PIN NO	SIGNALS	PIN NO	SIGNALS
1	P2C4	14	P2B1
2	P2C5	15	P2A6
3	P2C2	16	P2A7
4	P2C3	17	P2A4
5	P1C0	18	P2A5
6	P1C1	19	P2A2
7	P1B6	20	P2A3
8	P1B7	21	P2A0
9	P1B4	22	P2A1
10	P1B5	23	P2C6
11	P1B2	24	P2C7
12	P1B3	25	+5V
13	P1B0	26	GND

J2 PORTS CONNECTOR

PIN NO	SIGNALS	PIN NO	SIGNALS
1	P1C4	14	P1B1
2	P1C5	15	P1A6
3	P1C2	16	P1A7
4	P1C3	17	P1A4
5	P1C0	18	P1A5
6	P1C1	19	P1A2
7	P1B6	20	P1A3
8	P1B7	21	P1A0
9	P1B4	22	P1A1
10	P1B5	23	P1C6
11	P1B2	24	P1C7
12	P1B3	25	+5V
13	P1B0	26	GND

J3 PRINTER CONNECTOR

PIN NO	SIGNALS
1	STROBE *
2	DATA 0
3	DATA 1
4	DATA 2
5	DATA 3
6	DATA 4
7	DATA 5
8	DATA 6
9	DATA 7
10	NC
11	BUSY *
12	NC
13	NC

PIN NO	SIGNALS
14	NC
15	NC
16	NC
17	NC
18	GND
19	GND
20	GND
21	GND
22	GND
23	GND
24	GND
25	GND

J4 8751 USER EXPANSION CONNECTOR

PIN NO	SIGNALS
1	P0.0
3	P0.2
5	P0.4
7	P0.6
9	P1.0
11	P1.2
13	P1.4
15	P1.6
17	P3.0
19	P3.2/INT
21	P3.4
23	P3.6/WR*
25	P2.0
27	P2.2
29	P2.4
31	P2.6
33	ALE
35	NC
37	+5V
39	GND

PIN NO	SIGNALS
2	P0.1
4	P0.3
6	P0.5
8	P0.7
10	P1.1
12	P1.3
14	P1.5
16	P1.7
18	P3.1
20	P3.3
22	P3.5
24	P3.7/RD*
26	P2.1
28	P2.3
30	P2.5
32	P2.7
34	PSEN *
36	NC
38	+5V
40	GND

J5 SYSTEM EXPANSION AND TIMER CONNECTOR

PIN NO	SIGNALS	PIN NO	SIGNALS
1	P1.0	2	P1.1
3	P1.2	4	P1.3
5	P1.4	6	P1.5
7	P1.6	8	P1.7
9	P3.0	10	P3.1
11	P3.3	12	P3.4
13	P3.5	14	NC
15	BA0	16	BA1
17	BA2	18	BA3
19	BA4	20	BA5
21	BA6	22	BA7
23	BA8	24	BA9
25	BA10	26	BA11
27	CD0	28	CD1
29	CD2	30	CD3
31	CD4	32	CD5
33	CD6	34	CD7
35	CLK0	36	CLK1
37	CLK2	38	GATE 0
39	GATE1	40	GATE 2
41	OUT0	42	OUT 1
43	OUT2	44	OFFBOARDSEL*
45	BRD*	46	BWR*
47	+5V	48	+5V
49	GND	50	GND

J7 RS-232-C CONNECTOR

PIN NO	SIGNALS
1	GND
2	RXD*
3	TXD*
4	DTR
5	GND
6	DSR
7	RTS
8	CTS
9	NC

Note : If your terminal does not support hand-shaking signals loop RTS & CTS and DSR & DTR. Also remember ESA31 TXD should be connected to terminal RXD and so on.

CHAPTER 6

MONITOR ROUTINES ACCESSIBLE TO USER

ESA31 Monitor offers several user-callable routines both in the keyboard and serial modes of operation, details of which are given below. These routines can be used to considerably simplify the program development work.

NOTE: User should, as a general rule, save the registers of interest before calling the monitor routines and restore them after returning from the monitor routines.

6.1 KEYBOARD MONITOR ROUTINES ACCESSIBLE TO USER

Calling Address	Mnemonic	Function/Description
020BH	UPDAD	Updates address field of the display. The contents of the locations parlow (Internal memory locations 60H & 61H) are displayed in the address field. The contents of all the CPU registers flags are affected. If Reg. B=0 No dot, if B = 1,2,3,4 No. of dots in the address field
019BH	UPDDT	Updates data field of the display. The contents of the location parlow (Internal memory location 60H) are displayed in the data field. The contents of all CPU registers and flags are affected. If Reg. B=0, no dot, if Reg B = 1,2 No. of dots in the data field

0255H	OUTPUT	Outputs a string of 7 characters to display.The parameters for this routine are as follows: Reg DPTR = Starting address of string of characters. The string of characters will be displayed from program memory area or data memory area depending on the status of flag register bit F0. If flag register bit F0 = 0 program memory will be selected If flag register bit F0 = 1 data memory will be selected.
0170H	CLEAR	Clears the display. This routine blanks the entire display field
02A2H	RDKBD	Reads keyboard. This routine waits until a character is entered from the system keyboard and upon return, it places the character in the A register. The register A and F/F's are affected.

6.2 SERIAL MONITOR ROUTINES ACCESSIBLE TO USER

Calling Address	**Mnemonic**	**Function/Description**
16E2H	GETCH	Gets one character from the USART input parameters None.Output: A = Character(ASCII) received from USART.Reg. A, and flags are affected.
160EH	SOUTPT	Outputs one character to the USART. Inputs: A = Character (ASCII) to be output to USART. Reg. A, and flags are affected.
164BH	DISPM	Displays a string of characters. The string should be terminated by character Zero which is not output. Inputs:DPTR = Starting address of the string of characters. The string of characters will be displayed from program memory area or data memory area depending on the status of flag register bit F0. If flag register bit F0 = 0 program memory will be selected If flag register bit F0 = 1 data memory will be selected.

CHAPTER 7

AUDIO TAPE INTERFACE

7.1 INTRODUCTION

Audio tape is an economical, large capacity, non-volatile storage medium. During the program development, the user can store the necessary information (Programs and data) on the audio tape and at a later point of time, user can reload the saved information into the system memory. This relieves the user from the burden of manually entering large volume of information every time the system is powered on. Audio cassette recorder is an attractive solution as a back-up storage device, in comparison with the more versatile floppy disks, primarily because of the low cost. Any ordinary commercial tape recorder can be used.

The on-board audio tape interface consists of the necessary hardware and software to allow the user to store data and read data from a commercial audio tape recorder. The data is stored and read as named files.

7.2 INSTALLATION

The interface consists of two sockets - one marked as MIC and other as EAR. (Refer the Component Placement Diagram in Appendix A). Use any standard audio cable to connect MIC or EAR socket to the tape recorder.

7.2.1 OPERATIONAL HINTS

1) Connect only one socket MIC or EAR at any time to tape recorder. Connecting both the sockets permanently to the tape recorder may short circuit both the signal lines if the recorder common point connection is different from the convention followed in ESA31.

2) The volume and tone controls of your tape recorder may have to be adjusted to sufficiently high levels for reliable operation of this interface.

3) The reliability of the recording can be increased by using a recorder and tape of good quality.

4) The interface can accommodate normal variation in the tape speed. Ensure that your recorder does not produce too significant changes in the tape speed.

5) Avoid storing relatively large files. If the file is really large, split it into multiple files of smaller size. Though this scheme reduces the utilization of the tape space, it improves the chances of restoring the complete file correctly.

7.3 OPERATION FROM THE KEYBOARD MONITOR

The software allows the user to store data on the tape and read the recorded data from the tape, as named files.

7.3.1 STORING DATA ONTO TAPE

TPWR command implements this facility. Thus to invoke this facility, press TPWR key when prompted for command entry.

After typing TPWR, set the recorder in record mode and now enter the parameters as shown below:

Filename NEXT {PRG MEM| EXT DATA} starting address NEXT ending address EXEC

OPERATION

As soon as TPWR key is pressed, the display is cleared with a dot at the right edge of the address field. Enter the file name as a string of hexadecimal digits. Any number of digits can be entered, but the last 4 digits entered (currently displayed in the address field) are treated as valid. After entering the filename press the NEXT key. The display is cleared and a dot appears at the special field of the display. Now enter the type of memory by pressing PRG MEM or EXT DATA key to select program memory or data memory as the case may be. The display is again cleared with two dots at the right edge of the address field. Now enter the starting address of the block of data to be transferred to the tape and press the NEXT key. The display is again cleared with a dot in the address field. Now enter the ending address of the block of data to be transferred to the tape.

After entering the ending address, connect the microphone input of the recorder to the MIC jack of the tape interface hardware, press PLAY and REC keys of the recorder and then press the EXEC key.

Now data is transferred onto the tape. After writing the data, the system records a checksum byte also. During the transfer, the type of memory in the special field ,file name in the address field and ‘st” in the data field will be displayed to indicate that a file with the specific name is being stored onto tape. After transferring the data, control returns to the Monitor.

IMPORTANT NOTE

If the recorder is not ready and if you press the EXEC key, data would be sent out from the interface and this data would not be recorded on tape. Hence ensure that the recorder is ready and in record mode before pressing the EXEC key.

Example: To store the contents of locations 8000 - 80FF on tape with a file name 123F.

Key Pressed	**Display**			**Comments**
	Special Field	**Address Field**	**Data Field**	
RESET		-E S A	51	System Reset
TPWR		. . .		Tape write command
1		00.0.1.		
2		00.1.2.		
3		01.2.3		Filename 123F
F		1 2.3.F.		
NEXT	.			Type of memory
P	P	. .		Prompt for starting address
8	P	0 0 0.8.		
0	P	0 0 8.0.		
0	P	0 8 0.0.		
0	P	8 0 0.0.		Start address8000H
NEXT	P	.		Prompt for ending address
8	P	0 0 0 8.		
0	P	0 0 8 0.		
F	P	0 8 0 F.		
F	P	8 0 F F.		Ending address80FFH Connect the microphone of the recorder to MIC. Press REC and PLAY keys of the recorder.
EXEC	P	1 2 3 F S t		Data is being recorded on the tape Display is all dots.
		-		control returns to Monitor after completing the transfer.

7.3.2 READING DATA FROM TAPE

TPRD command implements this facility. Thus to invoke this facility, press TPRD key when prompted for command entry. After pressing TPRD, enter the desired filename followed by EXEC.Then set the recorder in PLAY mode.

TPRD Filename EXEC

OPERATION

As soon as TPRD key is pressed, the display is cleared with a dot at the right edge of the address field. The user must now enter the desired filename (As starting and ending addresses are already stored on the tape, there is no need for the user to enter these values).

Connect the earphone of the recorder to EAR jack of the Tape Interface Hardware. Position the tape well ahead of the approximate start of the desired file. Now press the EXEC key and then press the PLAY key of the recorder.

The display will be dashes in all the digit positions while the search is on for the desired file. As and when a filename is read by the system, it is displayed on the address field. When the specified file is found, the type of memory and the file name is displayed on special and address fields of the display. 'rd' will be displayed on the data field of the display indicating that data is being transferred from the tape on to the memory. As noted already, the starting and ending addresses of the memory block are read from the tape itself. After the data transfer, a checksum also, recorded on the tape during tape write, is read and is compared with the checksum of the actual data transferred. If they do not match, or if the data transfer is not correct, the system displays error message E r r and control returns to the monitor.If the data is transferred without any errors, control returns to the monitor which will display the command prompt.

EXAMPLE:

To read the contents of the file with the file name of 123F.

Key Pressed	**Display**			**Comments**
	Special Field	**Address Field**	**Data Field**	
RESET		- E S A	51	System Reset
TPRD		.		Tape Read command.
1		0 0 0 1.		
2		0 0 1 2.		
3		0 1 2 3.		
F		1 2 3 F.		Filename 123F (using earphone jack) to the EAR of

				the audio tape interface hardware. Press play of the recorder.
EXEC	-	- - - -	- -	Display will be dashes in address and data field, until the required file is found.
		X X X X		If any other file is found, its name will be displayed on the address field.
	P	1 2 3 F	r d	Specified file found Memory type file name will be displayed on special and address fields and rd will be displayed on the data field to indicate that the file is being read. Completion of data transfer successfully

7.4 OPERATION FROM THE SERIAL MONITOR

The software allows the user to store data on the tape and read the recorded data from the tape, as named files.

7.4.1 STORING DATA ONTO TAPE

“W” command implements this facility. The format for this command is as follows.

W {P|D} address1,address2,filename

Enter the following sequence when the system prompts for a command.

Type W, enter the type of memory, program memory or external data memory by entering P or D. Now enter the filename followed by<CR>. A valid filename consists of a sequence of hexadecimal digits. User may enter any number of digits, but the system retains only last four digits. If invalid filename is entered, the operation is aborted with an error message and control returns to monitor.

Ensure that the recorder is connected to the interface via MIC jack. Set up the recorder in record mode. The system will display the message “Writing data on to tape” and begin transfering the specified data onto tape, after the data a checksum byte is also written onto tape. After completing the transfer, the system will display the message

Finished writing onto tape
>

IMPORTANT NOTE

Data is sent out once the user enters the <CR> after the ending address. If the recorder is really not ready at this stage, the data is still sent out and this will not get recorded on the tape.

EXAMPLE: Store the contents of locations 8000H to 8FFFH of program memory with a filename 14B3, on the tape. Connect the recorder via the MIC jack and set it on RECORD mode. Enter the following command sequence.

>WP 8000 80FF 14B3 <CR>

Writing data onto tape

Finished writing onto tape

>

7.4.2 READING DATA FROM TAPE

L command implements this facility. The format for this command is as follows.

L filename <CR>

Type L, enter the filename and press <CR>

Now the system will display the message, “Searching File: filename”. Connect the recorder to the interface via the EAR jack. Position the tape well ahead of the approximate starting point of the desired file. Set the recorder in the play mode.

As and when the system finds a file, it displays the filename as shown below:

Filename X X X X

If the file is not the specified one,the search continues. When the specified file is found, it displays the message “Loading data from file” and begins loading data into memory.

As already noted, the start and end addresses of the memory block are read from the tape itself. Further, the checksum byte recorded on the tape during Tape Write operation, is also read. This checksum byte is compared with the checksum calculated from the actual data read from the tape and if they do not match, it is treated as error condition.

If the file is loaded without any errors, it displays the message, **“File loaded”** and control returns to the monitor. If any errors occur either in read or in writing into the memory, it displays the message, **“Unable to write at memory location :xxxx”** and control returns to the monitor.
(xxxx is the memory location)

Example: Read the contents of the file with filename 14B3.

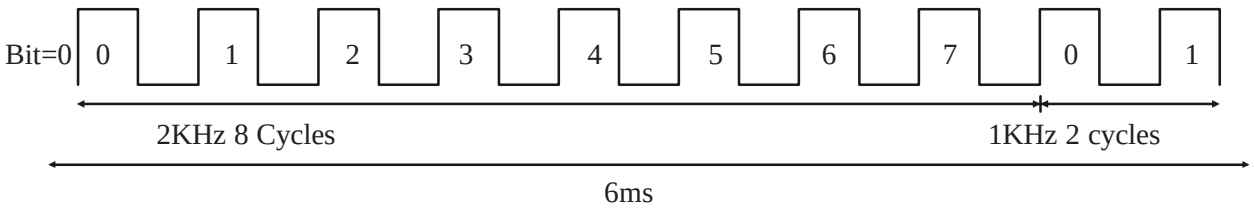
L 14B3 <<CR>>
Searching file: 14B3
Filename: 14B3
Loading data from file
File loaded

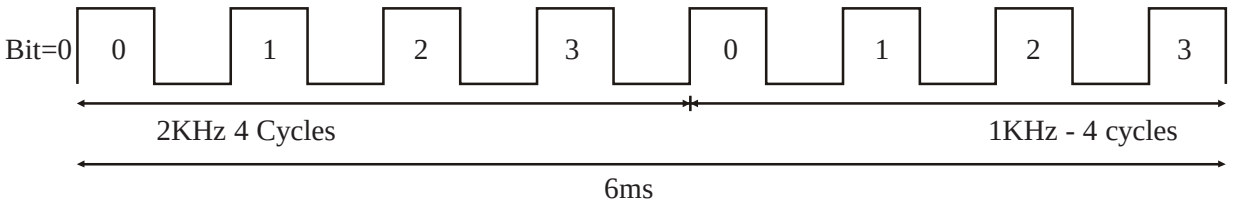
>

7.5 DATA RECORDING FORMATS

7.5.1 BIT FORMAT :

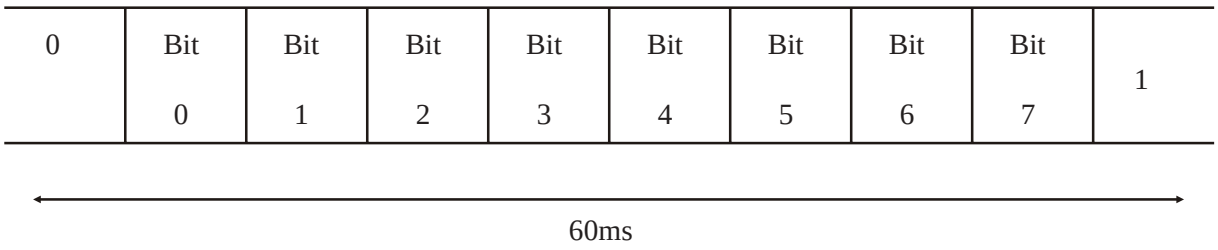
Both 0 and 1 are recorded as combination of some high frequency (2KHz) signals and some low frequency (1KHz) signals as shown below:





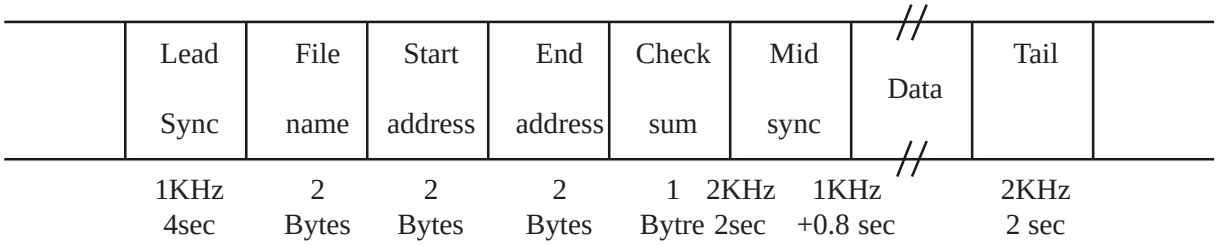
7.5.2 BYTE FORMAT:

A byte is recorded as one start bit (a zero), 8 data bits and one stop bit.



7.5.3 FILE FORMAT :

Before recording the file, a lead synchronization signal of 1 KHz frequency is recorded for 4 seconds. Then a seven byte header is recorded. The file header consists of filename (2 Bytes), start address (2 bytes), end address (2 bytes) and checksum (1 byte). Following this, a mid synchronization signal is recorded. This consists of a 2KHz signal for 2 sec and 1KHz signal for 0.8 secs. Now the file data is recorded. After this, a tail synchronization signal of 2KHz frequency is recorded for 2 seconds. Thus the file format is:



7.6 STORAGE CAPACITY

As is obvious from the above description of the data formats, the exact amount of data that can be stored on a tape depends upon the total number of files created. The larger the number, the smaller is the amount of data that can be stored (because of the space consumed by the larger number of synchronization signals). Assuming a typical number of 25 files for a standard tape of 90 minutes duration, we see that approximately 10 minutes (after allowing for about 15 seconds of inter-file gap during recording) may be taken up by synchronization signals. This allows us to record approximately 72 KB of data on one such cassette. However, if data is recorded in files of smaller size (say 128 or 256 bytes/file), the number of files stored will increase and total data storage capacity may fall to about 40K bytes.

CHAPTER 8

PARALLEL PRINTER INTERFACE

8.1 INTRODUCTION

ESA31 trainer supports centronics compatible parallel printer interface. The interface makes use of BUSY signal for hand-shaking and strobe pulses for synchronization. Using this facility the user can obtain hard copy on any centronics compatible printer. However to get properly formatted listing it is advisable to use 80/132 column printer. The on-board 8155A is made use of, to implement this interface.

8.2 INSTALLATION

To install the printer interface

a) Switch OFF the power supply.

b) Connect one end of the printer cable to J3 (25pin 'D' type connector) of ESA31. (Refer the component layout diagram in Appendix A to locate the connector J3).

c) Connect the other end of connector to the printer.

d) Configure the system for serial mode of operation by setting SW4 of the on-board DIP Switch to ON position.

e) Enable the printer interface by setting the SW5 of the on-board DIP Switch to ON position.

f) Switch on the power to the printer and ESA31.

NOTE: The necessary printer cable could be obtained from Electro Systems Associates (P) Ltd as an

optional accessory. However, the connector details are given in section 8.5 and the user can make use of these details to make a suitable cable if desired. Please note that the cable must be short enough to be driven by the on- board 8155A. We suggest a maximum length of 3 feet for reliable operation.

8.3 OPERATION

When the printer interface is installed and enabled as described above, any character sent to the console is sent to the printer also. For example, to obtain a hard copy of the contents of a block of memory locations, user can issue the D (Display Memory) command from the serial monitor. The contents of the specified memory block are printed exactly as they appear on the screen. Note that the D command itself is also printed.

NOTES:

1. All control and non-printable ASCII characters are printed as "." (ASCII Code 2EH).
2. If any errors occur during printing (for eg: printer is not in ON-LINE, paper-out error etc), the system will be looping indefinitely in the print character routine. To recover, user may have to press the RESET key.

8.4 DIRECT OUTPUT TO PRINTER

As already described, when the printer interface is enabled, any character sent to the console is sent to the printer also. This facility is available in the serial mode of operation only. However, user can directly access a routine "print character" to print a single character. This routine can be called from the user's program when the system is operating in either of the two modes-keyboard or serial. Further, this routine prints a character independent of the setting of SW5. Thus this routine can be used to print the desired information when the system is running in the keyboard mode. Even in the serial mode of operation, this routine can be used to print information which may not be sent to the console. The details of this routine are given below:

Name of the Routine:PRINT

Function: Print a character (Non-printable one is printed as ".").

Calling address: 1681H

Input: Register A = ASCII code of the character to be printed

Destroys: Registers A,B,DPTR,R0,Flags

Returns: Reg C = 1 if error otherwise C = 0

EXAMPLE: The following program, if entered and executed from the keyboard, prints 'ABCDEFGHIJKL'

Address	opcode		Mnemonic	Comment
8000	90 80 50		MOV DPTR,# MSG	;point to msg
8003	E4	L1:	CLR A	
8004	93		MOVC A, @A+DPTR	;get the chr
8005	60 0E		JZ L2	;end of msg?
8007	C0 82		PUSH DPH	;no, save pointer
8009	C0 83		PUSH DPL	
800B	12 16 81		LCALL PRINT	;print chr
800E	D0 83		POP DPL	;restore pointer
8010	D0 82		POP DPH	
8012	A3		INC DPTR	;point to next chr
8013	80 EE		SJMP L1	;repeat the process
8015	02 00 03	L2:	LJMP 0003	;save status and ;return to monitor
8050	41,42,43,44	DB :	41H, 42H, 43H, 44H	
8054	45,46,47,48		45H, 46H, 47H, 48H	
8058	49,4A,4B,4C		49H, 4AH, 4BH, 4CH	
805C	0D,0A,00		0DH, 0AH, 00H	

8.5 CONNECTOR DETAILS

The signal definitions on the 25-pin, female D type connector used for parallel printer interface are given below:

PIN NO ON J3	SIGNAL	DIRECTION FROM ESA 31	DESCRIPTION	PIN. NO. ON CENTRONICS CONNECTOR
1	STROBE	O/P	STROBE* pulse to the printer	1
2	Data 0	O/P	These signals represent 8	2
3	Data 1	O/P	bits of parallel data	3
4	Data 2	O/P	High - 1	4
5	Data 3	O/P	Low - 0	5
6	Data 4	O/P		6
7	Data 5	O/P		7
8	Data 6	O/P		8

PIN NO ON J3	SIGNAL	DIRECTION FROM ESA 31	DESCRIPTION	PIN. NO. ON CENTRONICS CONNECTOR
9	Data 7	O/P		9
10	BUSY	O/P		11
11	BUSY	I/P	A high incidates that printer can not receive data The Signals becomes high in following cases a) During the data entry b) During printing operation c) In the OFF - LINE state d) During printer error status	11
18-25	GND		Signal ground	19

CHAPTER 9

EPROM PROGRAMMER SYSTEM

9.1 INTRODUCTION

ESA31 EPROM programmer system is a powerful and easy to use facility provided by ESA31. This chapter describes the use of this EPROM Programmer System. The ESA31 trainer and the EPROM programmer interface module with a 26 core flat ribbon cable together form the EPROM Programmer system.

The system permits the user to program, verify, blank check and read any of the popular EPROMs 2716 through 27512. The system consists of the necessary hardware and software. The software can be invoked either from the keyboard monitor or from the serial monitor. A 28 Pin ZIF socket is provided for placing the EPROMs. When a 24-Pin EPROM is to be placed, it must be aligned with the bottom row i.e top two rows of ZIF are to be left blank.

The system uses Intelligent Programming Algorithm whenever possible which reduces the programming time significantly.

The devices supported by the system and the type number to be entered by the user are listed below:

Device		**Type number to be entered by the user**
2716	(@25V)	2716
27C16	(@25V)	2716
2732A	(@21V)	732A

Device	**Type number to be entered by the user**

27C32A	(@21V)	732A
27C32	(@25V)	2732
2732	(@25V)	2732
2764A	(@12.5V)	764A
27C64D	(@12.5V)	764A
27C64	(@21V)	2764
2764	(@21V)	2764
27128A	(@12.5V)	128A
27C128	(@12.5V)	128A
27128	(@21V)	0128
27256	(@12.5V)	0256
27C256	(@12.5V)	0256
27C256	(@21V)	2256
27512	(@12.5V)	0512

The device selection is totally software-controlled and no hardware changes or jumper settings are necessary for selecting any of the above listed devices.

9.2 INSTALLATION PROCEDURE

a) Turn OFF power to ESA31 trainer.

b) Attach the hardware module (EPROM Programmer Interface) to ESA 31 over connector J2 using 26 core ribbon cable supplied with the module.

c) Connect black, yellow and blue wires coming from the four pin polarised connector on the programmer module, to corresponding power supplies as shown below.

colour of the wire	**supply to be connected**
BLACK	GND
YELLOW	+12V
BLUE	+30V

d) Power ON the system

9.3. OPERATION FROM SERIAL MONITOR:

Enter P when prompted for command entry. Then system enters EPROM programmer menu and displays.

R: Read B: Blankcheck P: Program V: Verify E: Exit

Enter Option:

The Serial Monitor provides the following 4 commands to support the EPROM Programmer System:

P - Program command
V - Verify command
B - Blank check command
R - Read command

Enter the appropriate command, when prompted for command entry by the Serial Monitor.

Aborting a command:

With 'E' it exits from the EPROM Programmer Menu to the serial monitor prompt.

Once a specific command is issued, further prompts will depend on the command itself. However, if the user enters ESC whenever the system is looking for an entry from the user, the current operation is aborted and control returns to the warm start of the Serial Monitor. So in this case the user has to re-enter the EPROM Programmer menu by entering 'P' before issuing any command.

9.3.1 P COMMAND:

This command is used to program an EPROM. This command requires the following four parameters:

EPROM type	=	EPROM Type should be one of the types listed above in section 9.1
Buffer Start	=	Starting address of the source of data
Buffer End	=	Ending address of the source of data.
EPROMstart	=	Absolute starting address of the EPROM (from where programming is to begin)

NOTE: Buffer always means data memory area. If the program to be programmed on to an EPROM is in program memory area, the user has to transfer it to the data memory area before attempting programming. Similarly the user must note that when Read and Verify operations are performed, buffer means data memory area.

As soon as P is typed, the system displays each parameter value and prompts for new value. User can enter a new value followed by Carriage Return or simply enter Carriage Return if the displayed value is not to be changed.

Note that the parameters must satisfy certain conditions as listed below.

i) EPROM type can only be one of the valid types listed in section 9.1

ii) Buffer end address must be greater than or equal to the Buffer start address.

iii) The EPROM must have enough space to accomodate all the bytes specified by the Buffer start address and Buffer end address. In other words, the following relation must be satisfied.

EPROM Start + (Buffer end address-Buffer start address)
< = Highest absolute address of the EPROM.

For example, suppose EPROM type is 2764. Then its highest absolute address is 1FFF H. Suppose the other parameters are as follows:

Buffer Start = 8000
Buffer End = 9FFF
EPROM Start = 100

Then 100 + (9FFF-8000) 20FF> 1FFF. So this combination of parameters is invalid.

* After user enters the parameter values, the above mentioned constraints are checked and if any of the constraint is violated, it displays a message “invalid parameter(s) entered” and returns to the sub menu. However if the EPROM type entered is invalid, it immediately flashes a message “invalid EPROM type” and reprompts user to enter valid EPROM type.

After optional modification of the parameter values by the user,the system checks the EPROM for blank values (0FFH) in the required zone. It displays the message,

Blankcheck in progress...

If the EPROM is not blank, the following prompt appears:

EPROM is not blank @XXXX-YY
Continue programming (Y/N)

If user types N, the command is aborted and control returns to command prompt of the Serial Monitor.

If the user enters Y, the system proceeds further. Any other character results in error message and repetition of the same prompt.

* Now the following message appears:

Programming in progress ...

The system proceeds with programming and verification on a byte by byte basis. Intelligent Programming Algorithm is used if the EPROM can support it. This results in considerable reduction in programming time required for some devices.

If the complete programming is successful, the system will display a 16-bit checksum and control will return to the sub menu

If the programming is unsuccessful,the following information is displayed:

Programming failed @XXXX-YY

Where XXXX is the EPROM address where programming failed.and YY is the data at that location on EPROM.

NOTE: During programming and verification from serial mode, the location in EPROM and the corresponding data that is being programmed are displayed in the Keyboard/Display unit's display fields continuously.

After programming the specified range, the system displays 16 bit checksum and returns to sub menu.

Checksum = NNNN

9.3.2 V COMMAND

This command is used to verify the contents of an EPROM against a source.

The parameters and their interpretation is completely similar to that of the P Command.

If the verification is successful, the 16-Bit checksum is displayed and the control returns to EPROM programmer sub menu.

If the verification fails, a message and parameters at the point of failure are displayed as shown below.

Verification fails AAAA-BB CC-DDDD

Where AAAA is the EPROM address where verification has failed, BB is the data at that location. Similarly it also displays the corresponding buffer address (DDDD) and the data (CC) at that location. If there are multiple locations where verification has failed it will list out them in the same format as above. Then control returns to the EPROM programmer sub menu.The user can abort to main menu by pressing 'ESC' key

Thus the operation of this command is quite similar to the operation of the P command, except that here the EPROM is just verified, not programmed.

9.3.3 B COMMAND

This command is used to check if a specified range in the EPROM is blank (contains 0FFH)

This command requires the following three parameters:

Type : Same as for P command
EPROM Start : The absolute starting address of the EPROM
EPROM End : The absolute ending address of the EPROM.

The parameters must satisfy the following relations:

i) EPROM start <= Absolute last address of the EPROM

ii) EPROM End <= Absolute last address of the EPROM and

iii) EPROM End >= EPROM Start.

The parameter display and modification procedures are menu-driven and are similar to those of the P command.

The EPROM is checked for blank values in the specified range. The EPROM addresses and data read are displayed in the Keyboard/Display unit's display fields. If it is blank then the following message is displayed:

EPROM is blank

Then control returns to the sub menu.

If a location in the specified range is not blank, the following message is displayed.

EPROM is not blank @XXXX - YY

Where XXXX is the absolute EPROM address of the first non-blank location and YY is its content. Then it displays the address and data of subsequent non blank locations from next line onwards. If 'ESC' key is pressed control returns to the Serial Monitor.

9.3.4 R COMMAND

This command is used to transfer contents of the EPROM into the ESA31 memory space.

This command requires the following four parameters:

EPROM Type	:	same as for P command
EPROM Start	:	same as for B command
EPROM End	:	same as for B command
Buffer Start	:	starting address in ESA31 memory space.

The parameter display and modification procedures are menu driven and are completely similar to the ones described for P command.

The starting and ending addresses of the EPROM must satisfy the relations described for the B (Blank check) command.

After the optional modification of the parameters, the contents of the EPROM, in specified range are transfered into ESA31 memory, starting at the specified Buffer starting address. The EPROM addresses and data read are displayed in the Keyboard/Display unit's address and data fields respectively.

During the transfer, as each byte is written into memory, it is read back and verified. If the write is successful for all the locations, a 16-bit checksum is displayed and control returns to the sub menu.

If an error occurs during transfer (i.e unsuccessful write into a location), the following message is displayed.

Read fails @XXXX

where XXXX is the address of the location where write failure occured. Then control returns to the Serial Monitor.

9.4 OPERATION FROM THE KEYBOARD MONITOR

The Keyboard of ESA 31 has four keys which can be used to invoke the functions of the EPROM Programmer System. The keys and their functions are

Key	Function
PROG	Program an EPROM
VRFY	Verify an EPROM
BLNK	Blank check an EPROM
PRRD	Read an EPROM

Parameter Entry :

The following information is common to all the commands.

Whenever a parameter entry is required, the system will display a message in the data field which indicates the type of parameter to be entered and it will display the default value in the address field with a dot. To retain this value, user can press NEXT or EXEC, as required. Otherwise, user can enter the new value and then press NEXT or EXEC as required. All entries are made into the address field only. The different messages which can appear on the data field and their meanings are given below.

Message on the data field	Meaning
Pt	The type number of the EPROM
bS	Buffer Start Address
bE	Buffer End Address
PS	EPROM Starting Address
PE	EPROM Ending Address

All the parameter values are evaluated modulo 64K and a parameter entry is terminated by any valid delimiter. (Valid delimiters are: NEXT and EXEC). If an invalid entry is made, the message '-Err' is flashed in the address field and the prompt for the offending parameter is displayed again so that the user can enter the correct value. Pressing the 'NEXT' key takes the user to the next parameter if one is required - Otherwise it is treated as error condition.

Pressing the 'EXEC' key completes the parameter entry process of the command.

NOTE: Buffer means always data memory area. If the program to be programmed on to an EPROM is in program memory area, the user has to transfer it to the data memory area before attempting programming. Similarly the user must note that when Read and Verify operations are performed, buffer means data memory area.

9.4.1 PROG COMMAND:

This command is used to program an EPROM.

This command requires the following four parameters in that order.

EPROM Type	=	EPROM type (should be one of the types listed in section 9.1)
Buffer start	=	Starting address of the source of data.
BufferEnd	=	Ending address of the source of data.
EPROM Start	=	Absolute Starting address of the EPROM (from where programming is to begin).

* Note that these parameters must satisfy certain conditions as explained in the section 9.3.1 (P Command).
* Once the correct parameter values are available, the system checks the EPROM for blank values in the (FF) required zone. If the EPROM is blank in the required zone,the system proceeds further. Otherwise, it displays the message FULL in the address field and waits for user input. Now if the user presses 'EXEC' key, the system returns control to the Keyboard Monitor. If user presses "NEXT" key, the system proceeds further. Any other key will have no effect.
* After this, the system proceeds with programming and verification on a byte by byte basis. As each location in the EPROM gets programmed, the EPROM address is displayed in the address field and the programmed data is displayed in the data field.
* The system utilizes Intelligent Programming Algorithm wherever applicable and this reduces the programming time significantly.
* If the complete programming and verification is successful, the system will display a 12-bit checksum in the address field.

If the programming is unsuccessful, the address of the failed EPROM location is displayed in the address field and EPROM data is displayed in the data field. The system now waits for user input. If user presses "EXEC" key, control returns to the keyboard monitor. If "NEXT" key is pressed the next failed location and its content are displayed.

9.4.2 VRFY COMMAND

* This command is used to verify the contents of an EPROM against a source.
* The parameters and their interpretation is completely similar to that of the PROG command.
* If the verification is successful, the 12-bit checksum is displayed in the address field.

* If the verification fails, then “u” will be displayed in the special field.the parameters at the failed location are displayed as in the PROG command. (i.e EPROM address and data are displayed).

  Now if ‘EXEC’ key is pressed, control returns to the Monitor. If “NEXT” key is pressed, the next failed location and its content are displayed. Pressing any other key (except RESET) has no effect.

9.4.3. BLNK COMMAND.

* This command is used to check if a specified range in the EPROM is blank (contains FF).

This command requires the following three parameters in that order:

EPROM Type : Same as in PROG Command

EPROM Start : The absolute starting address of the EPROM

EPROM End : The absolute ending address of the EPROM

* The parameters must satisfy certain relations as explained in the section 9.4.3 on B command.
* Once valid parameter values are available, the EPROM is checked for blank state in the specified range.
* If the EPROM is blank in the specified range, the message -PAS is displayed in the address field. and control returns to the Keyboard Monitor. Otherwise, ‘b’ is displayed in the special field, the address of the first non-blank location is displayed in the address field and the EPROM data is displayed in the data field. The system now waits for user input. If “EXEC” is pressed, control returns to the monitor. If “NEXT” key is pressed, the next non blank location and its contents are displayed.

9.4.4 PRRD COMMAND

* This command is used to transfer the contents of the EPROM into the memory space of ESA 31.

This command requires the following four parameters in that order:

Type : Same as in PROG command

EPROM Start : Same as in Blank check command

EPROM End : Same as in Blank check command

BufferStart: : Starting address in ESA 31 memory space.

* These parameters must satisfy certain conditions as explained in the section 9.3.4 on R command.
* Once correct parameter values are available, the system reads the EPROM and transfers the contents to successive locations starting from the specified Buffer start. The EPROM address and data are displayed in address and data fields respectively.
* If the transfer is successful, a 12-bit checksum is displayed. Otherwise ‘r’ is displayed in the special field,the address of the offending location is displayed in the address field and EPROM data is displayed

in the data field. The system now waits for user input. If “EXEC” key is pressed, control returns to monitor. If “NEXT” key is pressed the next offending location and its data are displayed.

9.5 EXAMPLES

Example 1: From keyboard, read the contents of a 2732 EPROM into memory locations 8000H to 8FFFH.

Key Pressed	Display		comments
	Address Field	Data Field	
RESET	-E S A	51	
PRRD	2 7 3 2.	P t	Transfer command. Type prompt
NEXT	0 0 0 0.	P S	EPROM Start
NEXT	0 F F F.	P E	EPROM End
NEXT	0 0 0 0.	b S	Buffer start
8	0 0 0 8	b S	
0	0 0 8 0	b S	
0	0 8 0 0	b S	
0	8 0 0 0	b S	
EXEC	- 2 3 4	C S	Successful transfer. Checksum Return to monitor

In keyboard mode, if the user wants to know the 16 bit checksum (only 12 bit checksum is displayed on the trainer),the user may examine the external data memory locations E0E1H (MSB of the checksum is stored here) and E0E2H (LSB of the checksum is stored here).

Example 2 : From Serial monitor, program the contents of locations 8000H to 8FFFH into a 2764, starting at 1000H.

.P
R: Read B: Blankcheck P: Program V: Verify E: Exit
Enter option: P

EPROM type = 2732 - 2764 <CR>
Buffer start = 0000 - 8000 <CR>
Buffer End = 0000 - 8FFF <CR>
EPROM start = 0000 - 1000 <CR>

PROGRAMMING IN PROGRESS..
Checksum 1724

R: Read B: Blankcheck P: Program V: Verify E: Exit Enter option:E
>

CHAPTER 10

COMMUNICATION WITH A HOST COMPUTER SYSTEM

10.1 INTRODUCTION

As already noted, ESA31 operating in the serial mode, can be connected to a host computer system. When a computer system is the controlling element, it must be executing a driver software to communicate with ESA31.

ESA provides both WINDOWS and DOS based communication packages, which allow the user to establish a communication link between asynchronous serial ports of the computer (COM1/COM2), and ESA31. The CD supplied along with the trainer with the trainer contains these packages.

The communication software packages fully support the commands of ESA31. Further, they allow the contents of a disk file to be downloaded from the computer system into memory of ESA31. User can develop assembly language program on the PC, cross-assemble them using a suitable cross-assembler and linker to generate .HEX files that can be downloaded into ES 31 memory for execution. Thus the extensive developmenmt facilities available on the PC can be used to supplement the facilities available on ESA 31.

Further either of these packages allows uploading of data from memory of ESA31 to the computer. The data so uploaded is saved in a disk file. Thus this facility can conveniently be used to save user program as a .HEX file.

10.2 INSTALLATION

Configure ESA31 for serial mode of operation and set the serial port of ESA31 for desired baud and No parity (Refer sections 2.1.1 and 2.1.3). Then connect ESA31 to the host computer over the COM1/COM2

serial port using a suitable RS-232-C cable. (Refer to Technical Manual of your system for details regarding the signal definition on COM ports. The signal definitions of the RS-232-C port of ESA31 can be found in Appendix C)

i) Installation procedure for WIN31

Insert the CD containing the driver software into the available drive of the computer. To install the driver run SETUP.EXE file contained on the CD. The setup program will guide the user through the rest of the installation procedure. Once the software is installed successfully, the package offers a comprehensive online help for working with the commands.

ii) Installation procedure for XT31

Insert the CD containing the file XT31.EXE into the available drive and directly run the program XT31.EXE. This file may be directly copied to and executed from hard disk also.
Now the following message appears on the screen.

d) Now the following message appears on the screen.

XT31 Version 1.4

ELECTRO SYSTEMS ASSOCIATES PVT LTD

BANGALORE

ALT+S	-	Set Communication Parameters
CTRL+F1	-	Help
ALT+F1	-	Command Help
<Esc>	-	Clear Command
<F1>	-	Previous Command Character
<F3>	-	Command Recall
CTRL+U	-	Upload Command
CTRL+D	-	Download Command
! Command	-	Dos Shell/Command
ALT+X	-	P2 Exit
		Press any Key to Continue

e) XT31 checks for the presence of communication ports COM1 & COM2. If both ports are not available it will display the message **No serial port present as reported by BIOS** and exits to DOS. Otherwise XT31 will read the communication parameters from file “XT31.INS” and initializes the communication port. XT31 searches current directory for file “XT31.INS”. If search fails, it will search the path given by the DOS environment variable INIT. If the file is not present following message is displayed

XT31. INS is not found !
Serial parameters are set to COM1,9600, 8, 2, None
Do you want to change?
Yes No

If option “No” is selected the communication parameters: Serial Port COM1, Baud 9600, Data bits 8, Stop bits 2, Parity None are set. If option “Yes” is selected the communication parameters can be interactively modified as described in section 10.4.6 .

Now XT31 attempts to establish communication between the computer and ESA31. If successful the command prompt">" appears on the screen. (Subsequently during the power-on or RESET of the training, the sign-on message “ESA 8051 SERIAL MONITOR Vx.y. appears on the screen followed by command prompt “>”. The word “SERIAL” will be displayed on the trainer’s keyboard display. Otherwise it will display the message

Unable to transmit data
Retry or Ignore

If ESA31 is not powered on, power it on and press <R>to retry to establish the communication. If the sign-on message does not appear “SE” will be displayed on the trainer’s address field indicating that trainer is trying to communicate with the host. The above mentioned message appears on the host side again. Pressing <I>will exit XT31 to DOS. Now check for the following.

a) Ensure that ESA31 is connected to the correct COM port and that the COM port is functioning properly.

b) Ensure that ESA31 is functioning properly and configured correctly.

c) Check the RS-232-C cable and its connections.

Since the communciation package utilizes the hardware handshake signal DTR While communicating with ESA 31, the interfacing cable must support this signal also.

NOTE: XT31 utilizes an interrupt driven routine for reading characters from the COM port. Thus it is possible for XT31 to miss some characters if the system has any resident programs which are interrupt driven. (For example, many systems include a CLOCK program in the AUTOEXEC file, to display the time on the upper right corner of the screen). Hence it is desirable not to run any such resident programs while XT31 is running.

If the problem persists, please contact the manufacturer.

10.3 RETURNING TO DOS

User can terminate XT31 and return control to DOS by typing ALT+X when the program is waiting for a keyboard input.

10.4 OPERATIONAL DETAILS

The complete command set of the ESA31 is transparent and is fully supported by XT31 (Refer to chapter 4 for the serial monitor mode commands) Press F1 for the help command.

In addition, XT31 supports the file download, file upload and other commands which are explained below.

NOTE: During parameter entry, the system expects the alphabetic characters to be in upper case. Thus it is convenient to use the key board with the CAPS LOCK on.

10.4.1 DOWNLOAD OPERATION:

This feature allows downloading of the contents of an object code file into the memory of ESA31.

NOTE: The object code file must be a ".HEX" file with records in INTEL 8-Bit HEX format. Please refer to the relevant INTEL manuals for the definition of INTEL 8-Bit HEX format. Most of the cross assemblers for 8031 do produce object code files which are ".HEX" files with records in INTEL 8-Bit HEX format.

To perform download operation, type Ctrl+D in response to the command prompt (" > ").

The system will now prompt for the name of the disk file, from which the information is to be downloaded. The prompt is as follows:

Download filename [.HEX]:

Enter the file name with extension, terminated by < CR >. If the filename is invalid, it displays "FILE NOT FOUND" and prompts again for the filename. If the path specified is invalid, it displays a message "PATH NOT FOUND" and prompts again for the filename.If none of the above errors occur, the system will prompt for the Memory type as follows.

Memory type {P|D}:

The user has to enter P or D depending on whether program has to be downloaded to program or data memory.

Now the system will prompt for the starting address of the program as follows:

Start Address:

Enter the starting address followed by <CR>. A maximum of four hex digits are allowed for the starting address. Now the system will prompt for the ending address as follows:

End Address:

Enter the ending address followed by <CR>. A maximum of four hex digits are allowed for the ending address. Now the system will prompt for the load offset value as follows:

Load offset Address :

If the user wishes to download the file to an address range different from the actual address range of the file, then a suitable offset value can be entered to enable the file to be downloaded to the desired address range. For Example if the starting address of a file is 8000H and Ending address is 80FFH and if the user wishes to download it to an address range starting at C000H, the user has to enter an offset value of C000H-8000H= 4000H. In case the user wishes to download to the same address range an offset value of 0000 has to be entered or simply press <CR>. After obtaining the filename, starting address and the ending address, the system will read the file, gather the data in the specified address range, reformat the data for compatibility with the protocol required by ESA31 and send the data to ESA31.

While the downloading is going on the system will display the following message:

Downloading in progress...xxx

After downloading is over,the system returns the command prompt of ESA31. It also displays the starting address of each record being downloaded.

10.4.2 UPLOAD OPERATION:

This feature allows uploading of the data from the memory of ESA31 to the computer system and save the data in the specified disk file in INTEL 8-Bit HEX format.

To perform upload operation, type Ctrl +U in response to the command prompt (“>”).

The system will now prompt for the name of the disk file, into which the information is to be uploaded. The prompt is as follows:

UPLOAD FILENAME(.HEX)

Enter the file name with extension, terminated by <CR>. If the file already exists, then the system will display

File already exists
Overwrite?
Yes No Append

Select the first option by pressing <Y> to overwrite the contents of the existing file. Pressing <N> will let the user specify another file name. Select the third option <A> to append to the contents of the existing file.

If no error occurs, the system will prompt for the starting address of the program as follows :

Memory Type {P|D}

The user has to enter P or D depending on whether program has to be uploaded to program or data memory.

Now the system will prompt for the starting address of the program as follows:

Start Address

Enter the starting address terminated by <CR>. A maximum of four hex digits are allowed for the starting address.

Now the system will prompt for the ending address as follows:

End Address

Enter the ending address terminated by <CR>. A maximum of four hex digits are allowed for the ending address.

After obtaining the filename, starting address and the ending address, the system will gather the data in the specified address range of the ESA31, reformat the data into INTEL 8-Bit HEX records and store the data in the specified file.

Uploading in progress XXXX

While the uploading is going on, the system will display the starting address (XXXX) of each record being uploaded. Once the uploading is complete XT31 will let the user specify another address range. User may specify a new address range or enter <ESC> to terminate uploading operation.

The following error messages may appear during upload and download operations.

1. Invalid function number!
   - This is XT31 internal error, therefore contact ESA technical support for assistance.
2. File not found!
3. Path not found!
4. No more files!
5. Access denied!
6. Invalid file handle!
7. Insufficient disk space!
8. Unable to continue upload!
9. Colon is not present at the start of the record.!
10. Invalid data in (source file) the following record.!
11. Checksum error in the following Record!

10.4.3 DOS COMMANDS:

At the command prompt ">" any valid DOS command can be entered preceded by "!". XT31 environment is saved and the DOS command is executed. Then XT31 environment is restored and XT31 command prompt is displayed again.

10.4.4 BOTTOM LINE:

During the session XT31 displays many of the XT31 commands at the bottom line in reverse video for the convenience of user. The bottom line is displayed as

Ctrl+F1 Help, Alt+F1 CmdHlp, Alt+S Commn, <Esc> ClrCmd, Alt+X Exit, F1, F3, ↑, ↓

10.4.5 COMMAND RECALL:

This feature facilitates recall of the commands already entered by the user, upto a maximum of 16 commands. Press <F3> to recall the previous command. All the commands are kept in a circular buffer. User may use Up-arrow and Down- arrow keys to traverse through the sequence of commands in backward or forward direction respectively in a circular fashion. User may recall just the previous command, character by character by pressing <F1> desired number of times. Current command being entered can be cleared by using <ESC> key any time before pressing <CR>

10.4.6 COMMUNICATION:

comn port	int 1	Baud Rate	int 2	Data Bits	int 3	Stop Bits	int 4	Parity	int 5
COM 1	0	110	0	7	0	1	0	odd	0
COM 2	1	150	1	8	1	2	1	none	1
		300	2					even	2
		600	3						
		1200	4						
		2400	5						
		4800	6						
		9600	7						

Table 10.1

Communication parameters can be set during the session by pressing ALT+S. List of parameters and their current values will appear on the screen. Select the desired parameter with the help of arrow keys and keep the space-bar <sp> pressed till the desired value appears. The parameters allowed to be set are Communication Port (COM1/COM2), Baud Rate (110/ 150/ 300/ 600/ 1200/ 2400/ 4800/ 9600), Number of Data bits (7/8), Parity (NONE/ODD/EVEN) and Number of Stop bits (1/2). After selecting the desired values press <CR> to set the parameters or press <Esc> to ignore the values.

Communication parameters can also be modified,while user is in DOS by editing the file XT31.INS. This file contains single data line, having five integers separated by blanks, representing various communication parameters. These five integers represent serial communication port, baud rate, no. of data bits,no. of stop bits and parity, in sequence. Table 10.1 shows details of the integer values and corresponding parameters.

10.4.7 HELP:

On-line help is available on all ESA31 monitor commands and commands specific to XT31. Help facility can be selected by CTRL+F1. A menu of commands is displayed from which desired command can be selected by using arrow keys and help information about that command is displayed in the remaining part of the screen. Context sensitive help is available using ALT+F1. This facility can be used if help information is desired about the command being entered against command prompt.

CHAPTER 11

EXAMPLE PROGRAMS

11.1 Program to display ESA P LTD in trainer display. Execute the Program in KEYBOARD mode only.

OUTPUT EQU 0255H

ADDRESS	OBJECT	MNENOMIC	COMMENTS
8000	C2 D5	CLR 0D5	To select Program Memory.
8002	90 80 50	MOV DPTR, #8050	
8005	12 02 55	LCALL OUTPUT	Output routine to display string of 7 characters.
8008	02 80 00	LJMP 8000	

Keep data from 8050 to 8056 Program Memory.

8050 - D6
8051 - 77
8052 - 37
8053 - 83
8054 - 87
8055 - ED
8056 - 97

The program is in a continuous loop. Press RESET key to come out of the program.

11.2 Program to perform Multiplication of two numbers. Execute this program either in KEYBOARD or in SERIAL MODE. Taking 2 numbers in 9000 & 9001 Data Memory, and storing the result in 9002 & 9003 Data Memory.

ADDRESS	OBJECT	MNENOMIC		COMMENTS
8000	90 90 01	MOV	DPTR, #9001	Keep Data in 9000 & 9001 Data Memory location.
8003	E0	MOVX	A,@DPTR	
8004	F5 F0	MOV	0F0, A	
8006	90 90 00	MOV	DPTR, #9000	
8009	E0	MOVX	A,@DPTR	
800A	A4	MUL	AB	Perform Multiplication operation.
800B	90 90 02	MOV	DPTR, #9002	Store the result in 9002 & 9003 Data Memory
800E	F0	MOVX	@DPTR,A	
800F	A3	INC	DPTR	
8010	E5 F0	MOV	A,0F0	
8012	F0	MOVX	@DPTR,A	
8013	75 A0 E1	MOV	0A0,#0E1	
8016	78 02	MOV	R0,#02	
8018	E2	MOVX	A,@R0	
8019	20 E3 03	JB	0E3H,801F	Check for Dip Switch
801C	20 00 00	LJMP	0	Jump to SER Monitor
801F	02 04 FD	LJMP	04FD	Else to K.B. Prompt

11.3 Program to perform Division of 2 numbers. Execute the program either in KEYBOARD mode or in SERIAL mode. Taking 2 numbers in 9000 & 9001 Data Memory. After division operation store the result in 9002 & 9003 Data Memory.

ADDRESS	OBJECT	MNENOMIC		COMMENTS
8000	90 90 01	MOV	DPTR, #9001	Keep Data in 9000 & 9001 Data Memory location.
8003	E0	MOVX	A, @DPTR	
8004	F5 F0	MOV	0F0,A	

ADDRESS	OBJECT	MNENOMIC		COMMENTS
8006	90 90 00	MOV	DPTR,#9000	
8009	E0	MOVX	A,@DPTR	
800A	84	DIV	AB	Perform division operation.
800B	90 90 02	MOV	DPTR, #9002	Store the result in 9002 & 9003 Data Memory
800E	F0	MOVX	@DPTR,A	
800F	A3	INC	DPTR	
8010	E5 F0	MOV	A,0F0	
8012	F0	MOVX	@DPTR,A	
8013	75 A0 E1	MOV	0A0,#0E1	
8016	78 02	MOV	R0,#02	
8018	E2	MOVX	A,@R0	
8019	20 E3 03	JB	0E3H,801F	Check for Dip Switch
801C	02 00 00	LJMP	0	Jump to SER Monitor
801F	02 04 FD	LJMP	04FD	Else to be K.B. Prompt

11.4 Program to Display

ELECTRON SYSTEMS ASSOCIATES PVT LTD
BANGALORE on the console

Execute this program in SERIAL mode only.

	DISPM	**EQU**	**164BH**

ADDRESS	OBJECT	MNENOMIC		COMMENTS
8000	C2 05	CLB	005	
8002	90 80 50	MOV	DPTR, #8050	Keeping the Data in 8050 Program Memory.
8005	12 16 4B	LCALL	164B	Routine to display string of characters to console.
8008	02 00 00	LJMP	0	

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8050:	20	20	20	20	20	0A	20	20	20	20	20	45	4C	45	43	54
8060:	52	4F	20	53	59	53	54	45	4D	53	20	41	53	53	4F	43
8070:	49	41	54	45	53	20	50	56	54	20	4C	54	44	2E	0A	0D
8080:	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
8090:	20	20	20	20	42	41	4E	47	41	4C	4F	52	45	2E	0A	0D
80A0:		0A	0D	00												

11.5 Program to display ASCII Character on the console. Execute this program in serial mode only.

SOUTPUT EQU 160EH

ADDRESS	OBJECT	MNENOMIC		COMMENTS
8000	90 90 00	MOV	DPTR,#9000	Keep the ASCII value in 9000 data Memory.
8003	E0	MOVX	A,@DPTR	
8004	12 16 0E	LCALL	SOUTPUT	
8007	02 00 03	LJMP	3	; return to serial monitor

11.6 Program to convert ASCII to its HEX equivalent on the trainer display. If the code is less than 40, then 30 is subtracted from the code to get its binary equivalent. If the code is greater than 40, then the equivalent no. lies between A & F. Execute this program in KEYBOARD mode only.

ADDRESS	OBJECT	MNENOMIC		COMMENTS
8000	90 89 00	MOV	DPTR, # 8900	Keep the HEX eq in 8900 Data Memory
8003	E0	MOVX	A, @DPTR	
8004	F9	MOV	R1,A	
8005	C3	CLR	C	
8006	94 0A	SUBB	A,#0A	
8008	40 0B	JC	8015	Check for carry
800A	E9	MOV	A,R1	
800B	24 37	ADD	A,#37	
800D	78 60	MOV	R0,#60	Data field addess
800F	F6	MOV	@R0,A	
8010	12 01 9B	LCALL	091B	Routine to display in data field.
8013	80 EB	SJMP	8000	
8015	E9	MOV	A,R1	
8016	C3	CLR	C	
8017	24 30	ADD	A,#30	Add Accumulator content with 30.
8019	02 80 0D	LJMP	800D	

11.7 In the given byte checking the 5th bit is ‘1’ or ‘0’. If the 5th bit is ‘1’, 00 should be stored in 8901 Data Memory or if it is '0', ffh should be stored in 8901H data memory. Execute the program in KEYBOARD mode or in SERIAL mode.

ORG 8000H

ADDRESS	OBJECT	MNENOMIC		COMMENTS
8000	90 89 00	MOV	DPTR, #8900	Store the given byte in 8900 Data Memory
8003	E0	MOVX	A, @DPTR	
8004	A3	INC	DPTR	
8005	33	RLC	A	Rotate left three times.
8006	33	RLC	A	
8007	33	RLC	A	
8008	40 06	JC	8010	Check for carry
800A	74 FF	MOV	A,#0FF	Move FF in to accumulator
800C	F0	MOVX	@DPTR,A	
800D	02 80 13	LJMP	8013	
8010	74 00	MOV	A,#00	Move 00 in to Accumulator.
8012	F0	MOVX	@DPTR,A	
8013	75 A0 E1	MOV	0A0,#0E1	Check for DIP switch
8016	78 02	MOV	R0,#02	
8018	E2	MOVX	A,@R0	
8019	20 E3 03	JB	0E3,801F	
801C	02 00 03	LJMP	3	Jump to SFR monitor
801F	02 04 FD	LJMP	04FD	Else to K.B. Prompt

11.8 Program to display largest number among ‘N’ numbers. Execute the program either in KEYBOARD mode or in SERIAL mode.

ORG 8000H

PUTBYTE EQU 185EH
UPDDT EQU 019BH

ADDRESS	OBJECT	MNENOMIC		COMMENTS
8000	90 89 00	MOV	DPTR,#8900	The total no (‘N’) of data bytes is stored in 8900 data memory.

ADDRESS	OBJECT	MNENOMIC		COMMENTS
8003	E0	MOVX	A, @DPTR	
8004	FA	MOV	R2,A	Reg R2 is used as counter.
8005	90 89 01	MOV	DPTR,#8901	Data will be put from
8008	E0	MOVX	A,@DPTR	8901 onwards.
8009	1A	DEC	R2	Decrement counter.
800A	F9	MOV	R1,A	
800B	A3	INC	DPTR	Increment data memory.
800C	E0	MOVX	A,@DPTR	
800D	FB	MOV	R3,A	
800E	99	SUBB	A,R1	Comparing 2 numbers.
800F	50 14	JNC	8025	
8011	DA F8	DJNZ	R2,800B	
8013	75 A0 E1	MOV	0A0,#0E1	Check for DIP switch.
8016	78 02	MOV	R0,#02	
8018	E2	MOVX	A,@R0	
8019	20 E3 0E	JB	0E3,802A	
801C	E9	MOV	A,R1	
801D	F5 71	MOV	71H,A	Display the largest number on console.
801F	12 18 5E	LCALL	PUTBYTE	
8022	02 00 03	LJMP	03	
8025	EB	MOV	A,R3	
8026	F9	MOV	R1,A	
8027	02 80 11	LJMP	8011	
802A	E9	MOV	A,R1	
802B	78 60	MOV	R0,#60	Display the largest number on
802D	F6	MOV	@R0,A	trainer display.
802E	12 01 9B	LCALL	UPDDT	

8031 02 80 00 LJMP 8000

11.9 Program to display decimal count 0 to 20. Execute the program either in KEYBOARD mode or in SERIAL mode.

```
SOUTPT EQU 160EH
PUTBYTE EQU 185EH
UPDDT EQU 019BH
ORG 8000H
```

ADDRESS	OBJECT				MNENOMIC		COMMENTS
8000	75	A0	E1		MOV	0A0H,#0E1H	Check for DIP switch
8003	78	02			MOV	R0,#02H	
8005	E2				MOVX	A,@R0	
8006	20	E3	22		JB	0E3,KBD	
8009	7A	00			MOV	R2,#00H	Store count 00 in R2
800B	EA			RPT :	MOV	A,R2	
800C	F5	71			MOV	71H,A	
800E	12	18	5E		LCALL	PUTBYTE	Routine to display on console.
8011	7B	03			MOV	R3,#03H	
8013	74	08		LOOP:	MOV	A,#08H	
8015	12	16	0E		LCALL	SOUTPT	
8018	DB	F9			DJNZ	R3,LOOP	
801A	12	80	43		LCALL	DELAY	Delay routine.
801D	12	80	43		LCALL	DELAY	
8020	EA				MOV	A,R2	
8021	24	01			ADD	A,#01H	
8023	D4				DA	A	Perform Decimal Adjust.
8024	FA				MOV	R2,A	Accumulator operation.
8025	BA	21	E3		CJNE	R2,#21H,RPT	Compare the count with 21.
8028	02	00	00		LJMP	0000H	
802B	7A	00		KBD	MOV	R2,#00H	
802D	EA			RPT1:	MOV	A,R2	

ADDRESS	OBJECT				MNENOMIC		COMMENTS
802E	F5	60			MOV	60H,A	Routine to display.
8030	12	01	9B		LCALL	UPDDT	on Data field of trainer.
8033	12	80	43		LCALL	DELAY	
8036	12	80	43		LCALL	DELAY	
8039	0A				INC	R2	
803A	EA				MOV	A,R2	
803B	D4				DA	A	
803C	FA				MOV	R2,A	
803D	BA	21	ED		CJNE	R2,#21H,RPT1	
8040	02	04	FD		LJMP	04FDH	
8043	7C	FF		DELAY:	MOV	R4,#0FFH	Delay routine.
8045	7B	FF		BACK2:	MOV	R3,#0FFH	
8047	1B			BACK1:	DEC	R3	
8048	BB	00	FC		CJNE	R3,#00H,BACK1	
804B	1C				DEC4		
804C	BC	00	F6		CJNE	R4,#00H,BACK2	
804F	22				RET		

11.10 Program to display 24 hours digital clock in keyboard MODE. Execute the program from 8900. To change the Min, Hours change the data in the location 8903 and 8901 respectively.

```
UPDAD   EQU     020BH
UPDDT   EQU     019BH
```

ADDRESS	OBJECT				MNENOMIC		COMMENTS
8900	7D	23			MOV	R5,#23H	Store Hours in R5 Reg.
8902	7F	58			MOV	R7,#58H	Store Minutes in R7 Reg.
8904	ED			LOOP:	MOV	A,R5	
8905	F5	61			MOV	61H,A	
8907	EF				MOV	A,R7	
8908	F5	60			MOV	60H,A	
890A	12	02	0B		LCALL	UPDAD	Routine to Invoke Address field.

890D	79	00		UP :	MOV	R1,#00H	Store Seconds in R1 Reg.
890F	E9			RPT	MOV	A,R1	
8910	FE				MOV	R6,A	
8911	F5	60			MOV	60H,A	
8913	12	01	9B		LCALL	UPDDT	Routine to Invoke Data field.
8916	12	89	44		LCALL	DELAY	
8919	EE				MOV	A,R6	
891A	E9				MOV	A,R1	
891B	24	01			ADD	A,#01H	
891D	D4				DA	A	
891E	F9				MOV	R1,A	
891F	B9	60	ED		CJNE	R1,#60H,RPT	compare 60 sec is over or not.
8922	74	00			MOV	A,#00H	
8924	F5	60			MOV	60H,A	
8926	12	01	9B		LCALL	UPDDT	
8929	EF				MOV	A,R7	
892A	24	01			ADD	A,#01H	
892C	D4				DA	A	
892D	FF				MOV	R7,A	
892E	BF	60	26		CJNE	R7,#60H,MIN	Compare 60 Minutes is over or not.
8931	74	00			MOV	A,#00H	
8933	F5	60			MOV	60H,A	
8935	ED				MOV	A,R5	
8936	24	01			ADD	A,#01H	
8938	D4				DA	A	
8939	FD				MOV	R5,A	
893A	BD	24	23		CJNE	R5,#24H,HRS	Compare 24 hours is over or not.
893D	7D	00			MOV	R5,#00H	
893F	7F	00			MOV	R7,#00H	

8941	02	89	04		LJMP	LOOP	Jump to loop
8944	7A	04		DELAY:	MOV	R2,#04H	To provide 1 sec delay
8946	7C	FF		BACK3:	MOV	R4,#0FFH	
8948	7B	FF		BACK2:	MOV	R3,#0FFH	
894A	1B			BACK2:	DEC	R3	
894B	BB	00	FC		CJNE	R3,#00,BACK1	
894E	1C				DEC	R4	
894F	BC	00	F6		CJNE	R4,#00,BACK2	
8952	1A				DEC	R2	
8953	BA	00	F0		CJNE	R2,#00,BACK3	
8956	22				RET		
8957	EF			MIN:	MOV	A,R7	subroutine to display minutes.
8958	F5	60			MOV	60H,A	
895A	12	02	0B		LCALL	UPDAD	
895D	02	89	0D		LJMP	UP	
8960	ED			HRS:	MOV	A,R5	Subroutine to display hours
8961	F5	61			MOV	61H,A	
8963	12	02	0B		LCALL	UPDAD	
8966	7F	00			MOV	R7,#00H	
8968	02	89	0D		LJMP	UP	

11.11 Program to display 24 hours digital clock in SERIAL mode. Execute the program from 8000h. To change theMIN, HOURS change the data in location 8009h & 8007h respectively.

```
PUTBYTE EQU 185EH
SOUTPT EQU 160EH
DISPM EQU 164BH
```

ADDRESS	**OBJECT**				**MNENOMIC**		**COMMENTS**
8000	90	80	DD		MOV	DPTR,#DAT	Keep string 'HRS MIN SEC' in program memory.
8003	12	16	4B		LCALL	DISPM	dispaly routine.
8006	7F	23		START:	MOV	R7,#23H	Store hours in R7 reg.

8008	7E	58			MOV	R6,#58H	Stores Minutes in R6 Reg.
800A	EF			LOOP:	MOV	A,R7	
800B	F5	71			MOV	71H,A	
800D	12	18	5E		LCALL	PUTBYTE	Routine to put character on console.
8010	74	20			MOV	A,#20H	Provide space.
8012	12	16	0E		LCALL	SOUTPT	
8015	EE				MOV	A,R6	
8016	F5	71			MOV	71H,A	
8018	12	18	5E		LCALL	PUTBYTE	
801B	74	20			MOV	A,#20H	
801D	12	16	0E		LCALL	SOUTPT	
8020	7A	00		BEGIN:	MOV	R2,#00H	Keep seconds 00
8022	EA			SEC:	MOV	A,R2	In R2 reg.
8023	F5	71			MOV	71H,A	
8025	12	18	5E		LCALL	PUTBYTE	
8028	12	80	89		LCALL	DELAY	Delay routine.
802B	12	80	89		LCALL	DELAY	
802E	7B	03			MOV	R3,#03H	
8030	74	08		RPT:	MOV	A.#08H	
8032	12	16	0E		LCALL	SOUTPT	
8035	DB	F9			DJNZ	R3,RPT	
8037	EA				MOV	A,R2	
8038	24	01			ADD	A,#01H	
803A	D4				DA	A	
803B	FA				MOV	R2,A	
803C	BA	60	E3		CJNE	R2,#60H,SEC	Check for 60 sec over. or not.
803F	74	00			MOV	A,#00H	
8041	F5	71			MOV	71H,A	
8043	12	18	5E		LCALL	PUTBYTE	
8046	EE				MOV	A,R6	

```
8047    24  01          ADD   A,#01H
8049    D4              DA    A
804A    FE              MOV   R6,A
804B    BE  60  52      CJNE  R6,#60H,MIN   Check for 60 minutes over
                                            or not.
804E    7B  07          MOV   R3,#07H
8050    74  08      RPT1: MOV   A,#08H
8052    12  16  0E      LCALL SOUPT
8055    DB  F9          DJNZ  R3,RPT1
8057    74  00          MOV   A,#00H
8059    F5  71          MOV   71H,A
805B    12  18  5E      LCALL PUTBYTE
805E    7B  07          MOV   R3, #07H
8060    74  08      RPT2: MOV   A,#08H
8062    12  16  0E      LCALL SOUTPT
8065    DB  F9          DJNZ  R3, RPT2
8067    EF              MOV   A,R7
8068    24  01          ADD   A,#01H
806A    D4              DA    A
806B    FF              MOV   R7,A
806C    BF  24  52      CJNE  R7,#24H,HRS   check for 24 hours
                                            over or not.
806F    7B  04          MOV   R3,#04H
8071    74  08      RPT3: MOV A,#08H
8073    12  16  0E      LCALL SOUTPT
8076    DB  F9          DJNZ  R3, RPT3
8078    74  00          MOV   A,#00H
807A    F5  71          MOV   71H,A
807C    12  18  5E      LCALL PUTBYTE
807F    7F  00          MOV   R7,#00H
8081    7E  00          MOV   R6,#00H
8083    12  80  B7      LCALL CURSOR
```

```
8086        02  80  0A          LJMP    LOOP
8089 : Delay subroutine
8089        7D  C0      DELAY:  MOV     R5,#0C0H    This routine will provide
                                                    1sec delay.
808B        7C  FF      BACK2:  MOV     R4,#0FFH
808D        90  00  00          MOV     DPTR,#0H
8090        A3          GOBACK: INC     DPTR        Delay routine.
8091        E5  82              MOV     A,82H
8093        45  82              ORL     A,82H
8095        70  F9              JNZ     GOBACK
8097        1C          BACK1:  DEC     R4
8098        BC  00  FC          CJNE    R4,#00H,BACK1
809B        1D                  DEC     R5
809C        BD  00  EC          CJNE    R5,#00H,BACK2
809F        22                  RET
80A0 : Subroutine to display minutes.
80A0        7B  07      MIN:    MOV     R3,#07H
80A2        74  08      RPT4:   MOV     A,#08H
80A4        12  16  0E          LCALL   SOUTPT
80A7        DB  F9              DJNZ    R3,RPT4
80A9        EE                  MOV     A,R6
80AA        F5  71              MOV     71H,A
80AC        12  18  5E          LCALL   PUTBYTE
80AF        74  20              MOV     A,#20H
80B1        12  16  0E          LCALL   SOUTPT
80B4        02  80  20          LJMP    BEGIN
80B7 : Subroutine to move the cursor three times back.
80B7        7B  03      CURSOR: MOV     R3,#03H
80B9        74  08      RPT5:   MOV     A,#08H
80BB        12  16  0E          LCALL   SOUTPT
80BE        DB  F9              DJNZ    R3,RPT5
80C0        22                  RET
```

80C1 : Subroutine to display hours.

```
80C1        FF             HRS:  MOV    R7,A
80C2        F5  71                MOV    71H,A
80C4        12  18  5E            LCALL  PUTBYTE
80C7        74  20                MOV    A,#20H
80C9        12  16  0E            LCALL  SOUTPT
80CC        74  00                MOV    A,#00H
80CE        F5  71                MOV    71H,A
80D0        12  18  5E            LCALL  PUTBYTE
80D3        7E  00                MOV    R6,#00H
80D5        74  20                MOV    A,#20H
80D7        12  16  0E            LCALL  SOUTPT
80DA        02  80  20            LJMP   BEGIN
```

80DD : Subroutine to display HRS,MIN SEC on the console.

```
80DD    48 52 53 20 4D      DAT:  DB 48H, 52H, 53H, 20H, 4DH
80E2    49 4E 20 53 45            DB 49H, 4EH, 20H, 53H, 45H
80E7    43 0D 0A 00               DB 43H, 0DH, 0AH, 00H
80EB    02 80 06                  LJMP START
                                  END
```

11.12 Program to perform Addition of two numbers. This progran can be executed either in SERIAL mode or in KEYBOARD mode. Two numbers are taken from locations 9000 & 9001 of Data Memory. They are added and the result is stored in the location 9002 of Data Memory.

```
ADDR  OBJECT      LABLE    MNEMONIC                COMMENTS

                           SYMBOLS
8000              ORG 8000H
8000  90  90  00  START:  MOV   DPTR,#9000H   ;  KEEP DATA IN 9000
8003  E0                  MOVX  A,@DPTR          AND 9001 LOCATIONS
8004  F5  F0             MOV   0F0H,A           OF DATA MEMORY.
8006  90  90  01         MOV   DPTR,#9001H
8009  E0                  MOVX  A,@DPTR
800A  25  F0             ADD   A,0F0H       ;  ADD THEM.
800C  90  90  02         MOV   DPTR,#9002H  ;  STORE THE RESULT IN
800F  F0                 MOVX  @DPTR,A      ;  9002 LOCATION OF
8010  75  A0  E1         MOV   0A0H,#0E1H   ;  DATA MEMORY.
8013  78  02             MOV   R0,#02
```

```
8015    E2                      MOVX    A,@R0
8016    20    E3    03          JB      0E3H,L1     ;  READ DIP SWITCH
8019    02    00    00          LJMP    0           ;  JUMP TO SER. MONITOR
801C    02    04    FD    L1:   LJMP    04FDH       ;  ELSE TO K.B.PROMPT.
801F                            END
```

11.13 Program to perform subtraction of two numbers. This progran can be executed either in SERIAL mode or in KEYBOARD mode. Two numbers are taken from locations 9000 & 9001 of Data Memory. They are subtracted and the result is stored in the location 9002 of Data Memory.

```
ADDR    OBJECT          LABLE   MNEMONIC                COMMENTS

                                SYMBOLS
8000                    ORG     8000H
8000    90    90    01  START:  MOV     DPTR,#9001H     ;  KEEP DATA IN 9000
8003    E0                    MOVX    A,@DPTR         ;  AND 9001 LOCATIONS
8004    F5    F0              MOV     0F0H,A             OF DATA MEMORY.
8006    90    90    00        MOV     DPTR,#9000H
8009    E0                    MOVX    A,@DPTR
800A    95    F0              SUBB    A,0F0H          ;  SUBSTRACT THEM.
800C    90    90    02        MOV     DPTR,#9002H     ;  STORE THE RESULT IN
800F    F0                    MOVX    @DPTR,A         ;  9002 LOCATION OF
8010    75    A0    E1        MOV     0A0H,#0E1H      ;  DATA MEMORY.
8013    78    02              MOV     R0,#02
8015    E2                    MOVX    A,@R0
8016    20    E3    03        JB      0E3H,L1         ;  READ DIP SWITCH
8019    02    00    00        LJMP    0               ;  JUMP TO SER. MONITOR
801C    02    04    FD  L1:   LJMP    04FDH           ;  ELSE TO K.B.PROMPT.
801F                          END
```

APPENDIX A

COMPONENT PLACEMENT DIAGRAM

APPENDIX B

ASCII CODES

APPENDIX B-1

Hexadecimal	Decimal	Character	Hexadecimal	Decimal	Character
00	0	NUL	20	32	SP
01	1	SOH	21	33	!
02	2	STX	22	34	"
03	3	ETX	23	35	#
04	4	EOT	24	36	$
05	5	ENQ	25	37	%
06	6	ACK	26	38	&
07	7	BEL	27	39	,
08	8	BS	28	40	(
09	9	HT	29	41	)
0A	10	LF	2A	42	*
0B	11	VT	2B	43	+
0C	12	FF	2C	44	,
0D	13	CR	2D	45	-
0E	14	SO	2E	46	.
0F	15	SI	2F	47	/
10	16	DLE	30	48	0
11	17	DCI	31	49	1
12	18	DC2	32	50	2
13	19	DC3	33	51	3
14	20	DC4	34	52	4
15	21	NAK	35	53	5
16	22	SYN	36	54	6
17	23	ETB	37	55	7
18	24	CAN	38	56	8
19	25	EM	39	57	9
1A	26	SUB	3A	58	:
1B	27	ESC	3B	59	;
1C	28	FS	3C	60	<
1D	29	GS	3D	61	=
1E	30	RS	3E	62	>
1F	31	US	3F	63	?

APPENDIX B-2

Hexadecimal	Decimal	Character	Hexadecimal	Decimal	Character
40	64	@	60	96	,
41	65	A	61	97	a
42	66	B	62	98	b
43	67	C	63	99	c
44	68	D	64	100	d
45	69	E	65	101	e
46	70	F	66	102	f
47	71	G	67	103	g
48	72	H	68	104	h
49	73	I	69	105	i
4A	74	J	6A	106	j
4B	75	K	6B	107	k
4C	76	L	6C	108	l
4D	77	M	6D	109	m
4E	78	N	6E	110	n
4F	79	O	6F	111	o
50	80	P	70	112	p
51	81	Q	71	113	q
52	82	R	72	114	r
53	83	S	73	115	s
54	84	T	74	116	t
55	85	U	75	117	u
56	86	V	76	118	v
57	87	W	77	119	w
58	88	X	78	120	x
59	89	Y	79	121	y
5A	90	Z	7A	122	z
5B	91	[	7B	123	{
5C	92	/	7C	124	|
5D	93	]	7D	125	}
5E	94	^	7E	126	~
5E	95	-	7F	127	DEL

APPENDIX C

RS 232 C CABLE REQUIREMENT

APPENDIX C
RS 232 C CABLE REQUIREMENTS

THE ESA 31 REQUIRES A NULL MODEM CABLE IN ORDER TO COMMUNICATE WITH OTHER SYSTEMS.

THESE ARE THE CONNNECTIONS REQUIRED FOR THE NULL MODEM CABLE

Trianer Side	System Side
TXD Pin 3 >	< RXD (pin 3)
RXD Pin 2 >	< TXD (pin 2)
RTS Pin 7 >	< CTS (pin 5)
CTS Pin 8 >	< RTS (pin 4)
DSR Pin 6 >	< DTR (pin 20)
DTR Pin 4 >	< DSR (pin 6)
GND Pin 5 >	< GNS (pin 7)

Note :

1) Use male of 9 PIN 'D' Connector on ESA 31 side appropriate on other side.
2) If hardware handshaking is not required inter connect RTS and CTS (PIN 7 and 8) and DSR and DTR (PIN 6 and 4)

APPENDIX D

PRODUCT LIST

APPENDIX E

INSTRUCTION SET